Panduan Pengguna

AWS Identity and Access Management



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

AWS Identity and Access Management: Panduan Pengguna

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon adalah milik dari pemiliknya masing-masing, yang mungkin berafiliasi atau tidak berafiliasi dengan, terkait, atau disponsori oleh Amazon.

Table of Contents

Apa itu IAM? 1
Mengapa saya harus menggunakan IAM? 3
Akses bersama ke Anda Akun AWS 3
Izin granular 3
Akses aman ke AWS sumber daya untuk aplikasi yang berjalan di Amazon EC2 4
Multi-factor otentikasi (MFA) 4
Federasi identitas 4
Informasi identitas untuk jaminan 4
Kepatuhan PCI DSS 4
Kapan saya harus menggunakan IAM? 5
Ketika Anda melakukan fungsi pekerjaan yang berbeda 5
Ketika Anda diberi wewenang untuk mengakses AWS sumber daya 6
Saat Anda masuk sebagai pengguna IAM 6
Ketika Anda mengambil peran IAM 6
Saat Anda membuat kebijakan dan izin 7
Bagaimana cara mengelola IAM? 8
Gunakan Konsol Manajemen AWS 8
AWS Alat Baris Perintah 10
Gunakan AWS SDKs 12
Gunakan API Kueri IAM 12
Cara kerja IAM 13
Komponen permintaan 14
Bagaimana prinsipal diautentikasi 15
Dasar-dasar kebijakan otorisasi dan izin 16
..... 16
Membandingkan kredensial dan identitas IAM 18
Istilah 18
Perbedaan antara pengguna IAM dan pengguna di Pusat Identitas IAM 20
Menggabungkan pengguna dari sumber identitas yang ada 21
Metode yang berbeda untuk menyediakan akses pengguna 23
Mendukung akses pengguna terprogram 26
Bagaimana izin dan kebijakan menyediakan manajemen akses 27
Kebijakan dan akun 28
Kebijakan dan pengguna 28

Kebijakan dan grup IAM 29
Sesi dan peran pengguna gabungan 29
Kebijakan berbasis identitas dan berbasis sumber daya 30
Tentukan izin dengan otorisasi ABAC 31
Perbandingan ABAC dengan model RBAC tradisional 32
Memulai 34
Menyiapkan Akun AWS 35
Melihat Anda Akun AWS ID 37
Untuk melihat Akun AWS ID 37
Menggunakan alias untuk Anda Akun AWS ID 38
Membuat alias akun 39
Menghapus alias akun 40
Rencanakan akses ke AWS akun Anda 41
Sumber identitas 41
Manajemen akses 42
Kasus penggunaan untuk pengguna IAM 43
Gunakan otentikasi multi-faktor dengan identitas Anda 56
Bersiaplah untuk izin hak istimewa terkecil 57
Meninjau informasi terakhir yang diakses untuk akun Anda AWS 58
Menghasilkan kebijakan berdasarkan aktivitas akses 64
Menggunakan pencarian untuk menemukan sumber daya IAM 68
Praktik terbaik keamanan dan kasus penggunaan 71
Praktik terbaik keamanan 71
Memerlukan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara 72
Memerlukan beban kerja untuk menggunakan kredenSIAL sementara dengan peran IAM untuk mengakses AWS 73
Memerlukan otentikasi multi-faktor (MFA) 74
Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredenSIAL jangka panjang 75
Ikuti praktik terbaik untuk melindungi kredenSIAL pengguna root Anda 76
Terapkan izin hak istimewa terkecil 76
Memulai dengan AWS kebijakan terkelola dan bergerak menuju izin hak istimewa terendah 76
Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa terkecil berdasarkan aktivitas akses 77

Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredenSIAL yang tidak digunakan secara teratur 77
Gunakan ketentuan dalam kebijakan IAM untuk membatasi akses lebih lanjut 77
Verifikasi akses publik dan lintas akun ke sumber daya dengan IAM Access Analyzer 78
Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk izin yang aman dan fungsional 78
Buat pagar pembatas izin di beberapa akun 78
Menggunakan batas izin untuk mendelegasikan manajemen izin dalam akun 79
Praktik terbaik pengguna root 79
Amankan kredentif pengguna root Anda untuk mencegah penggunaan yang tidak sah 81
Gunakan kata sandi pengguna root yang kuat untuk membantu melindungi akses 81
Amankan login pengguna root Anda dengan otentikasi multi-faktor (MFA) 81
Jangan membuat kunci akses untuk pengguna root 82
Gunakan persetujuan multi-orang untuk masuk pengguna root jika memungkinkan 82
Gunakan alamat email grup untuk kredensional pengguna root 83
Batasi akses ke mekanisme pemulihan akun 83
Amankan Anda AWS Organizations kredenSIAL pengguna root akun 84
Memantau akses dan penggunaan 85
Kasus penggunaan bisnis 87
Pengaturan awal example corp 87
Gunakan kasus untuk IAM dengan Amazon EC2 88
Gunakan kasus untuk IAM dengan Amazon S3 90
Tutorial 93
Delegasikan akses di Akun AWS seluruh peran 93
Pertimbangan-pertimbangan 95
Prasyarat 96
Buat peran di Akun Tujuan 96
Berikan akses ke peran tersebut 100
Akses uji dengan mengalihkan peran 102
Sumber daya tambahan 107
Ringkasan 108
Buat kebijakan terkelola pelanggan 108
Prasyarat 109
Langkah 1: Buat kebijakan 109
Langkah 2: Lampirkan kebijakan 110
Langkah 3: Uji akses pengguna 111

Sumber daya terkait 111
Ringkasan 111
Gunakan kontrol akses berbasis atribut (ABAC) 111
Ikhtisar tutorial 112
Prasyarat 114
Langkah 1: Buat pengguna uji 115
Langkah 2: Buat kebijakan ABAC 117
Langkah 3: Buat peran 121
Langkah 4: Uji pembuatan rahasia 122
Langkah 5: Uji menampilkan rahasia 125
Langkah 6: Uji skalabilitas 127
Langkah 7: Uji memperbarui dan menghapus rahasia 129
Ringkasan 131
Sumber daya terkait 131
Menggunakan tanda sesi SAML untuk ABAC 132
Mengizinkan pengguna mengelola kredensial dan pengaturan MFA mereka 136
Prasyarat 137
Langkah 1: Buat kebijakan untuk menerapkan masuk MFA 138
Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda 139
Langkah 3: Uji akses pengguna Anda 140
Sumber daya terkait 142
Buat SAML IdP dengan CloudFormation 142
Prasyarat 142
Buat IdP 143
Verifikasi IdP 146
Hapus sumber daya 146
Detail templat 147
Templat 147
Buat peran federasi SAMP dengan CloudFormation 151
Prasyarat 152
Buat peran federasi 152
Uji peran federasi 154
Hapus sumber daya 155
Detail templat 155
Templat 156
Buat SAML IdP dan peran federasi dengan CloudFormation 162

Prasyarat ... 163
Buat sumber daya ... 163
Federasi uji ... 167
Hapus sumber daya ... 167
Detail templat ... 167
Templat ... 168
Identitas ... 178
Pengguna root IAM ... 179
Pengguna IAM: ... 179
Grup pengguna IAM ... 179
Peran IAM ... 179
Pengguna root akun AWS ... 179
Mengelola akses root secara terpusat untuk akun anggota ... 180
Memusatkan akses root ... 181
Melakukan tugas istimewa ... 185
MFA untuk pengguna root ... 192
Ubah kata sandi ... 200
Setel ulang kata sandi pengguna root yang hilang atau terlupakan ... 202
Buat kunci akses untuk pengguna root ... 204
Hapus kunci akses untuk pengguna root ... 207
Tugas yang memerlukan kredensial pengguna root ... 208
Sumber daya tambahan ... 211
Pengguna ... 211
Bagaimana AWS mengidentifikasi pengguna IAM ... 212
Pengguna dan kredenSIAL IAM ... 212
Pengguna dan izin IAM ... 214
Pengguna dan akun IAM ... 215
Pengguna IAM sebagai akun layanan ... 215
Cara pengguna IAM masuk AWS ... 215
Buat pengguna ... 220
Lihat pengguna IAM ... 222
Ganti nama pengguna ... 225
Menghapus pengguna ... 227
Kontrol akses pengguna ke konsol ... 231
Ubah izin pengguna ... 233
Kata sandi pengguna ... 241

Kelola kunci akses ... 262
Multi-factor otentikasi ... 292
Kredensi khusus layanan ... 372
Temukan kredensi yang tidak digunakan ... 383
Buat laporan kredensial ... 387
Kredensi IAM untuk CodeCommit ... 395
Kelola sertifikat server ... 398
Grup pengguna ... 404
Buat grup IAM ... 405
Lihat grup IAM ... 407
Mengedit pengguna dalam grup IAM ... 408
Melampirkan kebijakan ke grup pengguna ... 410
Ganti nama grup pengguna ... 411
Menghapus grup IAM ... 412
Peran ... 414
Kapan harus membuat pengguna IAM (alih-alih peran) ... 416
Istilah dan konsep peran ... 417
Sumber daya tambahan ... 420
Masalah confused deputy ... 421
Skenario umum ... 432
Penciptaan peran ... 446
Manajemen peran ... 514
Metode untuk mengambil peran ... 551
Penyedia identitas dan federasi ke AWS ... 713
Federasi dengan Pusat Identitas IAM ... 715
Federasi dengan IAM ... 715
Federasi dengan kumpulan identitas Amazon Cognito ... 716
Sumber daya tambahan ... 716
Skenario umum ... 716
Federasi OIDC ... 722
Federasi SAML 2.0 ... 748
Federasi AWS Identitas ke layanan eksternal ... 790
Kasus penggunaan umum ... 791
Cara Kerjanya ... 792
Memulai dengan federasi identitas keluar ... 793
Memahami klaim token ... 799

Mengontrol akses dengan kebijakan IAM ... 805
Kredensial keamanan sementara ... 807
AWS STS and AWS daerah ... 807
Skenario umum untuk kredensial sementara ... 808
Contoh aplikasi yang menggunakan kredensial sementara ... 810
Sumber daya tambahan untuk kredensial sementara ... 811
Bandingkan AWS STS kredensialnya ... 811
Token pembawa layanan ... 815
Minta kredentif keamanan sementara ... 816
Gunakan kredenSIAL sementara dengan AWS sumber daya ... 828
Izin untuk kredensial keamanan sementara ... 833
Kelola AWS STS dalam sebuah Wilayah AWS ... 872
AWS STS Wilayah dan titik akhir ... 876
Aktifkan akses konsol broker identitas kustom ... 881
Tag untuk sumber daya IAM ... 897
Pilih AWS konvensi penamaan tag ... 898
Aturan untuk menandai di IAM dan AWS STS ... 900
Menandai pengguna IAM ... 902
Kata kunci IAM rolls ... 906
Tandai kebijakan yang dikelola pelanggan ... 909
Menandai OIDC identity provider ... 912
Kata kunci IAM SAML identity provider ... 915
Menandai profil instans ... 918
Sertifikat server tag ... 920
Menandai perangkat MFA virtual ... 923
Tandai instans manajer akses akun ... 926
Lulus tag sesi ... 928
Manajemen akses ... 945
Sumber daya manajemen akses ... 946
Kebijakan dan Izin ... 947
Jenis kebijakan ... 947
Kebijakan dan pengguna root ... 955
Gambaran dari kebijakan JSON ... 955
Berikan hak akses paling rendah ... 960
Kebijakan terkelola dan kebijakan inline ... 962
Perimeter data ... 973

Batas izin 981
Identitas vs sumber daya 995
Kontrol akses menggunakan kebijakan 998
Kontrol akses ke pengguna dan peran IAM menggunakan tag 1011
Kontrol akses ke AWS sumber daya menggunakan tag 1014
Akses sumber daya lintas akun 1020
Teruskan sesi akses 1027
Contoh kebijakan 1030
Kelola kebijakan IAM 1114
Sumber daya tambahan 1115
Buat kebijakan IAM 1115
Validasi kebijakan 1126
Pengujian kebijakan 1127
Menambah atau menghapus izin identitas 1141
Peningkatan versi IAM 1154
Edit kebijakan IAM 1158
Hapus kebijakan IAM 1169
Memperbaiki izin menggunakan informasi akses 1175
Ringkasan kebijakan 1763
Ringkasan kebijakan (daftar layanan) 1764
Tingkat akses dalam ringkasan kebijakan 1775
Ringkasan layanan (daftar tindakan) 1778
Ringkasan tindakan (daftar sumber daya) 1784
Contoh ringkasan kebijakan 1789
Izin diperlukan untuk mengakses sumber daya IAM 1799
Izin untuk mengurus identitas IAM 1799
Izin untuk bekerja di Konsol Manajemen AWS 1801
Berikan izin di seluruh AWS rekening 1802
Izin satu layanan untuk mengakses layanan lainnya 1802
Tindakan yang diperlukan 1803
Contoh kebijakan untuk IAM 1804
Delegasi sementara 1808
Gambaran umum 1808
Cara kerja delegasi sementara 1809
Memulai permintaan delegasi sementara 1811
Tinjau permintaan delegasi sementara 1814

Cabut akses delegasi sementara 1816
Mengelola Izin untuk Permintaan Delegasi 1817
Notifikasi 1818
CloudTrail 1819
Delegasi sementara untuk Mitra 1820
Gambaran umum 1820
Cara kerjanya 1809
Kualifikasi mitra 1821
Proses onboarding 1822
Kuesioner mitra 1823
Memahami integrasi Anda 1824
Memahami izin 1828
Pedoman evaluasi kebijakan 1830
Templat kebijakan 1833
Membangun integrasi Anda 1844
Manajer akses akun 1853
Apa itu manajer akses akun? 1853
Mengapa menggunakan manajer akses akun? 1853
Kemampuan utama 1855
Memulai 1856
Prasyarat dan pertimbangan 1857
Aktifkan manajer akses akun 1857
Memperbarui firewall dan gateway 1858
Hapus manajer akses akun 1860
Kelola akses ke Akun AWS 1861
Administrator yang didelegasikan 1861
Siapkan peran IAM Anda 1865
Tetapkan dan hapus akses 1867
Melihat atau mencari tugas 1871
Sesi dan kontrol akses 1873
Akses tenaga kerja 1874
Portal pengguna 1874
Memulai dengan portal akses akun 1875
Akses pengguna ke Akun AWS 1876
Keamanan 1881
Identitas dan manajemen akses untuk manajer akses akun 1881

Perlindungan data di manajer akses akun ... 1889
Pencatatan dan pemantauan di manajer akses akun ... 1890
Kuota ... 1890
Kuota penugasan kelompok ... 1890
Batas throttle ... 1890
Contoh kode ... 1892
IAM ... 1899
Hal-hal mendasar ... 1905
Skenario ... 2696
AWS STS ... 3189
Hal-hal mendasar ... 3190
Skenario ... 3222
Keamanan ... 3475
AWS kredensi keamanan ... 3476
Pertimbangan keamanan ... 3477
Akses terprogram ... 3479
AWS pedoman audit keamanan ... 3483
Kapan melakukan audit keamanan ... 3484
Pedoman untuk audit ... 3484
Tinjau kredensi AWS akun Anda ... 3484
Tinjau pengguna IAM Anda ... 3485
Tinjau grup IAM Anda ... 3485
Tinjau IAM role Anda ... 3486
Tinjau penyedia IAM untuk SAML and OpenID Connect (OIDC) Anda ... 3486
Tinjau aplikasi seluler Anda ... 3486
Tips untuk meninjau kebijakan IAM ... 3487
Perlindungan data ... 3489
Enkripsi data dalam IAM dan AWS STS ... 3490
Manajemen kunci dalam IAM dan AWS STS ... 3490
Privasi lalu lintas internetwork di IAM dan AWS STS ... 3490
Pencatatan log dan pemantauan ... 3491
Log acara dengan CloudTrail ... 3492
Lacak tugas istimewa di CloudTrail ... 3519
Validasi kepatuhan ... 3523
Ketahanan ... 3524
Praktik terbaik untuk ketahanan IAM ... 3526

Keamanan infrastruktur 3526
Konfigurasi dan analisis kerentanan 3527
AWS kebijakan yang dikelola 3527
IAMReadOnlyAccess 3528
IAMUserChangePassword 3528
IAMAccessAnalyzerFullAccess 3528
IAMAccessAnalyzerReadOnlyAccess 3530
AccessAnalyzerServiceRolePolicy 3531
IAMAuditRootUserCredentials 3532
IAMCreateRootUserPassword 3534
IAMDeleteRootUserCredentials 3535
S3UnlockBucketPolicy 3537
SQSUnlockQueuePolicy 3538
AccountAccessManagerServiceRolePolicy 3539
Pembaruan kebijakan 3539
Fitur keamanan di luar IAM 3546
Penganalisis Akses IAM 3548
Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal 3548
Mengidentifikasi akses internal ke sumber daya penting bisnis 3551
Mengidentifikasi akses yang tidak digunakan yang diberikan kepada pengguna dan peran IAM 3552
Memvalidasi kebijakan terhadap AWS praktik terbaik 3552
Memvalidasi kebijakan terhadap standar keamanan yang Anda tentukan 3553
Menghasilkan kebijakan 3553
Harga untuk IAM Access Analyzer 3553
Temuan 3554
Bagaimana temuan bekerja 3556
Memulai dengan IAM Access Analyzer 3561
Dasbor temuan 3579
Tinjau temuan 3583
Filter temuan 3586
Mengarsipkan temuan 3594
Selesaikan temuan 3594
Temuan kesalahan 3598
Jenis sumber daya yang mendukung 3603
Administrator terdelegasi 3612

Aturan arsip ... 3614
Pemantauan EventBridge dengan ... 3616
Integrasi CSPM Security Hub ... 3628
Logging dengan CloudTrail ... 3637
Tombol filter ... 3640
Menggunakan Peran Terkait Layanan ... 3655
Akses pratinjau ... 3657
Mempratinjau akses di konsol Amazon S3 ... 3658
Mempratinjau akses dengan IAM Access Analyzer APIs ... 3659
Memeriksa untuk memvalidasi kebijakan ... 3663
Cara kerja pemeriksaan kebijakan khusus ... 3664
Contoh kebijakan referensi untuk memeriksa akses baru ... 3665
Memeriksa pemeriksaan kebijakan kustom yang gagal ... 3666
Validasi dengan pemeriksaan kebijakan dasar ... 3666
Referensi pemeriksaan kebijakan ... 3669
Validasi dengan pemeriksaan kebijakan khusus ... 3814
Pembuatan kebijakan Penganalisis Akses IAM ... 3816
Cara kerja pembuatan kebijakan ... 3816
Informasi tingkat tindakan dan layanan ... 3817
Hal yang perlu diketahui ... 3817
Izin diperlukan ... 3819
Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (konsol) ... 3822
Buat kebijakan menggunakan AWS CloudTrail data di akun lain ... 3826
Buat kebijakan berdasarkan CloudTrail aktivitas (AWS CLI) ... 3830
Buat kebijakan berdasarkan CloudTrail aktivitas (AWS API) ... 3830
Layanan pembuatan kebijakan IAM Access Analyzer ... 3831
Kuota Penganalisis Akses IAM ... 4388
Memecahkan masalah IAM ... 4391
Saya tidak bisa masuk ke AWS akun ... 4391
Saya kehilangan access key saya ... 4391
Variabel kebijakan tidak berfungsi ... 4392
Perubahan yang saya buat tidak selalu langsung terlihat ... 4392
Saya tidak berwenang untuk melakukan: iam: DeleteVirtualMFADevice ... 4393
Bagaimana cara membuat pengguna IAM dengan aman? ... 4394
Sumber daya tambahan ... 4395
Akses pesan kesalahan ditolak ... 4395

Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan . 4396
Saya mendapatkan “akses ditolak” ketika saya membuat permintaan dengan kredensial keamanan sementara 4398
Contoh akses ditolak 4399
Akses pesan kesalahan ditolak dengan ID otorisasi (Pratinjau) 4407
Gambaran umum 4407
Konsep dan definisi 4408
Memecahkan masalah menggunakan AWS Konsol Manajemen 4409
Memecahkan masalah menggunakan AWS API atau AWS CLI 4410
Saat Anda tidak menerima ID otorisasi 4410
Pertimbangan tambahan 4411
Layanan yang didukung 4411
Masalah pengguna root 4411
Kebijakan IAM 4413
Atasi masalah menggunakan editor visual 4414
Memecahkan masalah dengan ringkasan kebijakan 4419
Pemecahan masalah manajemen kebijakan 4430
Memecahkan masalah pada dokumen kebijakan JSON 4430
Kunci Sandi dan Kunci Keamanan FIDO 4437
Saya tidak dapat mengaktifkan kunci keamanan FIDO saya 4438
Saya tidak bisa masuk menggunakan kunci keamanan FIDO saya 4439
Saya kehilangan atau merusak kunci keamanan FIDO saya 4439
Masalah lainnya 4439
Peran IAM 4439
Saya tidak dapat mengsumsikan peran 4440
Peran baru muncul di AWS akun 4442
Saya tidak dapat mengedit atau menghapus peran di Akun AWS 4443
Saya tidak berwenang untuk melakukan: iam: PassRole 4443
Mengapa saya tidak bisa mengambil peran dengan sesi 12 jam? (AWS CLI, AWS API) 4444
Saya menerima kesalahan saat mencoba beralih peran di konsol IAM 4444
Peran saya memiliki kebijakan yang memungkinkan saya melakukan tindakan, tetapi saya mendapatkan “akses ditolak” 4445
Layanan tidak membuat versi kebijakan default peran tersebut 4445
Tidak ada kasus penggunaan untuk peran layanan di konsol 4447
IAM dan Amazon EC2 4448

Ketika saya mencoba meluncurkan instance, saya tidak melihat peran dalam daftar Peran IAM konsol Amazon EC2 4448
Kredensial pada instans saya untuk peran yang salah 4449
Saat saya mencoba memanggil `AddRoleToInstanceProfile`, saya mendapatkan kesalahan `AccessDenied` 4449
Amazon EC2: Ketika saya mencoba meluncurkan instance dengan peran, saya mendapatkan kesalahan `AccessDenied` 4450
Saya tidak dapat mengakses kredensial keamanan sementara di instans EC2 saya 4450
Apa arti kesalahan dari dokumen `info` di subtree IAM? 4451
IAM dan Amazon S3 4452
Bagaimana saya memberikan akses anonim ke bucket Amazon S3? 4453
Saya masuk sebagai Akun AWS pengguna root. Mengapa saya tidak dapat mengakses bucket Amazon S3 di bawah akun saya? 4453
Federasi SAML 2.0 4453
Respons SAML tidak valid 4454
RoleSessionName diperlukan 4455
Tidak diotorisasi untuk AssumeRoleWithSAML 4455
RoleSessionName Karakter tidak valid 4456
Karakter Identitas Sumber Tidak Valid 4456
Tanda tangan respons tidak valid 4457
Kunci pribadi tidak valid 4457
Gagal menghapus kunci pribadi 4457
Gagal menghapus kunci pribadi karena ID Kunci tidak cocok dengan kunci pribadi 4458
Gagal mengambil peran 4458
Tidak dapat mengurai metadata 4458
Tidak dapat memperbarui penyedia identitas 4459
Tidak dapat mengatur mode enkripsi pernyataan ke Diperlukan karena tidak ada kunci pribadi yang disediakan 4459
Tidak dapat menambah dan menghapus kunci pribadi dalam permintaan yang sama 4459
Penyedia yang ditentukan tidak ada 4460
DurationSeconds melebihi MaxSessionDuration 4460
Batas kunci pribadi 2 tercapai 4460
Respons tidak berisi audiens yang dibutuhkan 4460
Bagaimana IAM bekerja dengan layanan lain AWS 4461
Buat sumber daya IAM dengan CloudFormation 4461
IAM dan template CloudFormation 4462

Pelajari lebih lanjut tentang CloudFormation 4462
Gunakan AWS CloudShell dengan IAM 4463
Dapatkan izin IAM untuk AWS CloudShell 4463
Berinteraksi dengan IAM 4464
Bekerja dengan AWS SDK 4466
Referensi 4468
Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARN) 4468
Format ARN 4468
Cari format ARN untuk sumber daya 4470
Jalur di ARN 4470
Pengidentifikasi IAM 4472
Nama dan jalur yang ramah 4472
ARN IAM 4473
Pengidentifikasi unik 4481
Memahami prefiks ID yang unik 4482
Mendapatkan pengidentifikasi unik 4483
IAM dan kuota AWS STS 4484
Persyaratan nama IAM 4484
Kuota objek IAM 4485
Kuota Penganalisis Akses IAM 4487
Kuota IAM Roles Anywhere 4487
Kuota permintaan STS 4487
Batas karakter IAM dan STS 4488
Dukungan titik akhir tumpukan ganda 4493
Dukungan titik akhir tumpukan ganda IAM 4494
AWS STS dukungan titik akhir dual-stack 4494
Titik akhir VPC antarmuka 4495
Ketersediaan titik akhir VPC 4496
Buat titik akhir VPC untuk IAM 4497
Buat titik akhir VPC untuk AWS STS 4498
Buat titik akhir VPC untuk AWS STS Penemuan OIDC 4499
Kontrol akses ke AWS STS dengan kebijakan titik akhir VPC 4502
Gunakan tombol kondisi VPC untuk mengontrol akses federasi 4510
Layanan yang bekerja dengan IAM 4516
Layanan yang bekerja dengan IAM 4518
Informasi selengkapnya 4546

AWS Tanda Tangan Versi 4 4550
Bagaimana AWS SigV4 berfungsi 4551
Bagaimana AWS SigV4a berfungsi 4552
Kapan menandatangani permintaan 4552
Mengapa permintaan ditandatangani 4553
Sumber daya tambahan 4553
Elemen permintaan SIGv4 4554
Metode autentikasi 4557
Buat permintaan yang ditandatangani 4565
Minta contoh tanda tangan 4579
Memecahkan masalah SIGv4 4581
Referensi kebijakan 4586
Referensi elemen JSON 4587
Logika evaluasi kebijakan 4701
Tata bahasa kebijakan 4727
AWS kebijakan terkelola untuk fungsi pekerjaan 4736
Kunci kepatuhan global 4754
Kunci kondisi IAM 4840
Tindakan, sumber daya, dan kunci kondisi 4894
Sumber daya 4895
Identitas 4895
Kredensial (RDS, access key, dan perangkat MFA) 4895
Izin dan kebijakan 4896
Federasi dan delegasi 4896
IAM dan lainnya AWS produk 4897
Menggunakan IAM dengan Amazon EC2 4897
Menggunakan IAM dengan Amazon S3 4897
Menggunakan IAM dengan Amazon RDS 4897
Menggunakan IAM dengan Amazon DynamoDB 4898
Praktik kerahasiaan umum 4898
Sumber daya umum 4898
Membuat permintaan kueri HTTP 4900
Titik akhir 4901
HTTPS diperlukan 4901
Menandatangani permintaan API IAM 4901
Riwayat dokumen 4903

..... mmmmcmxxxv

Apa itu IAM?

Follow us on Twitter

AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman. Dengan IAM, Anda dapat mengelola izin yang mengontrol AWS sumber daya mana yang dapat diakses pengguna. Anda menggunakan IAM untuk mengontrol siapa yang dapat terautentikasi (masuk) dan berwenang (memiliki izin) untuk menggunakan sumber daya. IAM menyediakan infrastruktur yang diperlukan untuk mengontrol autentikasi dan otorisasi untuk Akun AWS Anda.

Untuk petunjuk tentang cara mengatur Akun AWS dan membuat identitas Anda, lihatMemulai dengan IAM.

Identitas

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang disebut pengguna Akun AWS root yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Untuk tugas yang memerlukan kredensial pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Gunakan IAM untuk menyiapkan identitas lain selain pengguna root Anda, seperti administrator, analis, dan pengembang, dan beri mereka akses ke sumber daya yang mereka butuhkan untuk berhasil dalam tugas mereka.

Manajemen akses

Setelah pengguna diatur di IAM, mereka menggunakan kredentif masuk mereka untuk melakukan otentikasi. AWS Otentikasi disediakan dengan mencocokkan kredentif masuk ke prinsipal (pengguna IAM, prinsipal AWS STS federasi, peran IAM, atau aplikasi) yang dipercaya oleh. Akun AWS Selanjutnya, permintaan dibuat untuk memberikan akses kepada principal ke sumber daya. Akses diberikan sebagai tanggapan atas permintaan otorisasi jika pengguna telah diberi izin ke sumber daya. Misalnya, saat pertama kali masuk ke konsol dan berada di halaman Beranda konsol, Anda tidak mengakses layanan tertentu. Ketika Anda memilih layanan, permintaan otorisasi dikirim ke layanan tersebut dan layanan akan melihat apakah identitas Anda ada dalam daftar pengguna yang diberi otorisasi, kebijakan apa yang diberlakukan untuk mengontrol tingkat akses yang diberikan, dan kebijakan lain yang mungkin berlaku. Permintaan otorisasi dapat dibuat oleh kepala sekolah di dalam diri Anda Akun AWS atau dari orang lain Akun AWS yang Anda percayai.

Setelah diberi otorisasi, principal dapat mengambil tindakan atau melakukan pengoperasian pada sumber daya di Akun AWS Anda. Misalnya, kepala sekolah dapat meluncurkan Amazon Elastic Compute Cloud instance baru, memodifikasi keanggotaan grup IAM, atau menghapus Amazon Simple Storage Service bucket.

> **Tip**
>
> AWS Skill Builder menyediakan pengantar video 10 menit untuk IAM: Pengantar AWS Identity and Access Management di situs web AWS Skill Builder.

Ketersediaan layanan

IAM, seperti banyak AWS layanan lainnya, pada akhirnya konsisten. IAM mencapai ketersediaan tinggi dengan mereplikasi data di beberapa server di dalam pusat data Amazon di seluruh dunia. Jika permintaan untuk mengubah beberapa data berhasil, perubahan tersebut akan dilakukan dan disimpan dengan aman. Namun, perubahan harus direplikasi di seluruh IAM, yang dapat memakan waktu. Perubahan tersebut mencakup membuat atau memperbarui pengguna, grup, peran, atau kebijakan. Kami sarankan Anda tidak memasukkan perubahan IAM seperti itu di jalur kode penting dengan ketersediaan tinggi di aplikasi Anda. Sebaliknya, buat perubahan IAM dalam inisialisasi terpisah atau rutinitas pengaturan yang lebih jarang Anda lakukan. Selain itu, pastikan untuk memverifikasi bahwa perubahan telah dibuat merata sebelum alur kerja produksi bergantung padanya. Untuk informasi selengkapnya, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

Informasi biaya layanan

AWS Identity and Access Management (IAM), AWS IAM Identity Center dan AWS Security Token Service (AWS STS) adalah fitur AWS akun Anda yang ditawarkan tanpa biaya tambahan. Anda hanya dikenakan biaya jika Anda mengakses AWS layanan lain menggunakan pengguna IAM atau kredentif keamanan AWS STS sementara.

Analisis akses eksternal IAM Access Analyzer ditawarkan tanpa biaya tambahan. Namun, Anda akan dikenakan biaya untuk analisis akses yang tidak digunakan dan pemeriksaan kebijakan pelanggan. Untuk daftar lengkap biaya dan harga untuk IAM Access Analyzer, lihat harga IAM Access Analyzer.

Untuk informasi tentang harga AWS produk lain, lihat halaman harga Amazon Web Services.

Integrasi dengan AWS layanan lain

IAM terintegrasi dengan banyak AWS layanan. Untuk daftar AWS layanan yang bekerja dengan IAM dan fitur IAM dukungan layanan, lihat. AWS layanan yang bekerja dengan IAM

Mengapa saya harus menggunakan IAM?

AWS Identity and Access Management adalah alat yang ampuh untuk mengelola akses ke AWS sumber daya Anda dengan aman. Salah satu manfaat utama menggunakan IAM adalah kemampuan untuk memberikan akses bersama ke AWS akun Anda. Selain itu, IAM memungkinkan Anda menetapkan izin granular, memungkinkan Anda mengontrol tindakan apa yang dapat dilakukan pengguna berbeda pada sumber daya tertentu. Tingkat kontrol akses ini sangat penting untuk menjaga keamanan AWS lingkungan Anda. IAM juga menyediakan beberapa fitur keamanan lainnya. Anda dapat menambahkan otentikasi multi-faktor (MFA) untuk lapisan perlindungan ekstra, dan memanfaatkan federasi identitas untuk mengintegrasikan pengguna dengan mulus dari jaringan perusahaan Anda atau penyedia identitas lainnya. IAM juga terintegrasi dengan AWS CloudTrail, menyediakan pencatatan terperinci dan informasi identitas untuk mendukung persyaratan audit dan kepatuhan. Dengan memanfaatkan kemampuan ini, Anda dapat membantu memastikan bahwa akses ke AWS sumber daya penting Anda dikontrol dan aman dengan ketat.

Akses bersama ke Anda Akun AWS

Anda dapat memberikan izin kepada orang lain untuk mengelola dan menggunakan sumber daya dalam akun AWS Anda yang tanpa harus membagikan kata sandi atau access key Anda.

Izin granular

Anda dapat memberikan izin yang berbeda kepada orang yang berbeda untuk sumber daya yang berbeda. Misalnya, Anda mungkin mengizinkan beberapa pengguna akses lengkap ke Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, dan layanan lainnya. AWS Untuk pengguna lain, Anda dapat mengizinkan akses read-only ke beberapa bucket Amazon S3, atau izin untuk mengelola hanya beberapa instans Amazon EC2, atau untuk mengakses informasi penagihan Anda tetapi tidak ada yang lain.

Akses aman ke AWS sumber daya untuk aplikasi yang berjalan di Amazon EC2

Anda dapat menggunakan fitur IAM untuk memberikan kredensial secara aman untuk aplikasi yang berjalan pada instans EC2. KredenSIAL ini memberikan izin bagi aplikasi Anda untuk mengakses AWS sumber daya lain. Contohnya termasuk bucket S3 dan tabel DynamoDB.

Multi-factor otentikasi (MFA)

Anda dapat menambahkan autentikasi dua faktor ke akun Anda dan ke pengguna individu untuk keamanan ekstra. Dengan MFA, Anda atau pengguna Anda harus memberikan tidak hanya kata sandi atau kunci akses untuk menangani akun Anda, tetapi juga kode dari perangkat yang dikonfigurasi secara khusus. Jika Anda sudah menggunakan kunci keamanan FIDO dengan layanan lain, dan memiliki konfigurasi yang AWS didukung, Anda dapat menggunakan WebAuthn untuk keamanan MFA. Untuk informasi selengkapnya, lihat Konfigur asi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Federasi identitas

Anda dapat mengizinkan pengguna yang sudah memiliki kata sandi di tempat lain—misalnya, di jaringan perusahaan Anda atau dengan penyedia identitas internet—untuk mengakses kata sandi Anda. Akun AWS Pengguna ini diberikan kredentif sementara yang sesuai dengan rekomendasi praktik terbaik IAM. Menggunakan federasi identitas meningkatkan keamanan AWS akun Anda.

Informasi identitas untuk jaminan

Jika menggunakan AWS CloudTrail, Anda menerima catatan log yang memuat informasi tentang pihak yang meminta sumber daya di akun Anda. Informasi tersebut didasarkan pada identitas IAM.

Kepatuhan PCI DSS

IAM mendukung pemrosesan, penyimpanan, dan transmisi data kartu kredit oleh pedagang atau penyedia layanan, dan telah divalidasi sebagai sesuai dengan Standar Keamanan Data (DSS) Industri Kartu Pembayaran (PCI). Untuk informasi selengkapnya tentang PCI DSS, termasuk cara meminta salinan Paket Kepatuhan AWS PCI, lihat PCI DSS Level 1.

Kapan saya harus menggunakan IAM?

AWS Identity and Access Management adalah layanan infrastruktur inti yang menyediakan dasar untuk kontrol akses berdasarkan identitas di dalamnya AWS. Anda menggunakan IAM setiap kali Anda mengakses AWS akun Anda. Cara Anda menggunakan IAM akan tergantung pada tanggung jawab spesifik dan fungsi pekerjaan dalam organisasi Anda. Pengguna AWS layanan menggunakan IAM untuk mengakses AWS sumber daya yang diperlukan untuk pekerjaan sehari-hari mereka, dengan administrator memberikan izin yang sesuai. Administrator IAM, di sisi lain, bertanggung jawab untuk mengelola identitas IAM dan menulis kebijakan untuk mengontrol akses ke sumber daya. Terlepas dari peran Anda, Anda berinteraksi dengan IAM setiap kali Anda mengotentikasi dan mengotorisasi akses ke sumber daya. AWS Ini bisa melibatkan masuk sebagai pengguna IAM, mengambil peran IAM, atau memanfaatkan federasi identitas untuk akses tanpa batas. Memahami berbagai kemampuan IAM dan kasus penggunaan sangat penting untuk mengelola akses aman ke AWS lingkungan Anda secara efektif. Dalam hal membuat kebijakan dan izin, IAM menyediakan pendekatan yang fleksibel dan terperinci. Anda dapat menentukan kebijakan kepercayaan untuk mengontrol prinsipal mana yang dapat mengambil peran, selain kebijakan berbasis identitas yang menentukan tindakan dan sumber daya yang dapat diakses pengguna atau peran. Dengan mengonfigurasi kebijakan IAM ini, Anda dapat membantu memastikan bahwa pengguna dan aplikasi memiliki tingkat izin yang sesuai untuk melakukan tugas yang diperlukan.

Ketika Anda melakukan fungsi pekerjaan yang berbeda

AWS Identity and Access Management adalah layanan infrastruktur inti yang menyediakan dasar untuk kontrol akses berdasarkan identitas di dalamnya AWS. Anda menggunakan IAM setiap kali Anda mengakses AWS akun Anda.

Cara Anda menggunakan IAM berbeda, tergantung pada pekerjaan yang Anda lakukan. AWS

- Pengguna layanan — Jika Anda menggunakan AWS layanan untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredentif dan izin yang Anda butuhkan. Saat Anda menggunakan fitur yang lebih canggih untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda.
- Administrator layanan — Jika Anda bertanggung jawab atas sumber AWS daya di perusahaan Anda, Anda mungkin memiliki akses penuh ke IAM. Adalah tugas Anda untuk menentukan fitur dan sumber daya IAM mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM.

- Administrator IAM — Jika Anda seorang administrator IAM, Anda mengelola identitas IAM dan menulis kebijakan untuk mengelola akses ke IAM.

Ketika Anda diberi wewenang untuk mengakses AWS sumber daya

Otentikasi adalah cara Anda masuk AWS menggunakan kredentif identitas Anda. Anda harus diautentikasi sebagai Pengguna root akun AWS, pengguna IAM, atau dengan mengambil peran IAM.

Anda dapat masuk sebagai identitas federasi menggunakan kredenSIAL dari sumber identitas seperti AWS IAM Identity Center (Pusat Identitas IAM), otentikasi masuk tunggal, atau kredenSIAL. Google/ Facebook Untuk informasi selengkapnya tentang cara masuk, lihat Cara masuk ke Akun AWS Anda dalam Panduan Pengguna AWS Sign-In .

Untuk akses terprogram, AWS menyediakan SDK dan CLI untuk menandatangani permintaan kriptografis. Untuk informasi selengkapnya, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Saat Anda masuk sebagai pengguna IAM

Pengguna IAM adalah identitas dengan izin khusus untuk satu orang atau aplikasi. Sebaiknya gunakan kredensial sementara alih-alih pengguna IAM dengan kredensial jangka panjang. Untuk informasi selengkapnya, lihat Memerlukan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara di Panduan Pengguna IAM.

Grup IAM menentukan kumpulan pengguna IAM dan mempermudah pengelolaan izin untuk pengguna dalam jumlah besar. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Ketika Anda mengambil peran IAM

Peran IAM adalah identitas dengan izin khusus yang menyediakan kredensial sementara. Anda dapat mengambil peran dengan beralih dari pengguna ke peran IAM (konsol) atau dengan memanggil operasi AWS CLI AWS API. Untuk informasi selengkapnya, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM berguna untuk akses pengguna terfederasi, izin pengguna IAM sementara, akses lintas akun, akses lintas layanan, dan aplikasi yang berjalan di Amazon EC2. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Saat Anda membuat kebijakan dan izin

Anda memberikan izin kepada pengguna dengan membuat kebijakan, yang merupakan dokumen yang mencantumkan tindakan yang dapat dilakukan pengguna dan sumber daya yang dapat memengaruhi tindakan tersebut. Tindakan atau sumber daya apa pun yang tidak diizinkan secara eksplisit ditolak secara default. Kebijakan dapat dibuat dan dilampirkan ke prinsipal (pengguna, grup pengguna, peran yang diasumsikan oleh pengguna, dan sumber daya).

Anda dapat menggunakan kebijakan ini dengan peran IAM:

- Kebijakan kepercayaan — Mendefinisikan prinsi pal mana yang dapat mengambil peran, dan dalam kondisi apa. Kebijakan kepercayaan adalah jenis kebijakan berbasis sumber daya tertentu untuk peran IAM. Peran hanya dapat memiliki satu kebijakan kepercayaan.
- Identity-based kebijakan (sebaris dan terkelola) — Kebijakan ini menentukan izin yang dapat dilakukan pengguna peran (atau ditolak untuk melakukan), dan sumber daya mana.

Gunakan Contoh kebijakan berbasis identitas IAM untuk membantu Anda menentukan izin untuk identitas IAM Anda. Setelah menemukan kebijakan yang Anda butuhkan, pilih lihat kebijakan untuk melihat JSON untuk kebijakan tersebut. Anda dapat menggunakan dokumen kebijakan JSON sebagai contoh untuk kebijakan Anda sendiri.

> **Note**
>
> Jika Anda menggunakan Pusat Identi IAM klasik untuk mengelola pengguna, Anda menetapkan kumpulan izin di Pusat Identitas IAM alih-alih melampirkan kebijakan izin ke prinsipal. Saat Anda menetapkan kumpulan izin ke grup atau pengguna di Pusat Identitas AWS IAM, Pusat Identitas IAM membuat peran IAM yang sesuai di setiap akun, dan melampirkan kebijakan yang ditentukan dalam kumpulan izin ke peran tersebut. Pusat Identitas IAM mengelola peran, dan mengizinkan pengguna resmi yang telah Anda tetapkan untuk mengambil peran tersebut. Jika Anda mengubah kumpulan izin, Pusat Identitas IAM memastikan bahwa kebijakan dan peran IAM terkait diperbarui sesuai.
> Anda juga dapat menggunakan manajer akses akun — fitur IAM yang memungkinkan Anda menetapkan peran IAM yang ada yang Anda buat dan kelola ke pengguna dan grup Pusat Identitas IAM, memberi Anda akses ke set fitur peran IAM lengkap. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.
> Untuk informasi selengkapnya tentang Pusat Identitas IAM, lihat Apa itu Pusat Identitas IAM? dalam Panduan AWS IAM Identity Center Pengguna.

Bagaimana cara mengelola IAM?

Mengelola AWS Identity and Access Management dalam suatu AWS lingkungan melibatkan pemanfaatan berbagai alat dan antarmuka. Metode yang paling umum adalah melalui Konsol Manajemen AWS, antarmuka berbasis web yang memungkinkan Anda melakukan berbagai tugas administratif IAM, mulai dari membuat pengguna dan peran hingga mengonfigurasi izin.

Untuk pengguna yang lebih nyaman dengan antarmuka baris perintah, AWS menyediakan dua set alat baris perintah - AWS Command Line Interface dan. AWS Tools for Windows PowerShell Ini memungkinkan Anda untuk mengeluarkan perintah terkait IAM langsung dari terminal, seringkali lebih efisien daripada menavigasi konsol. Selain itu, AWS CloudShell memungkinkan Anda menjalankan perintah CLI atau SDK langsung dari browser web Anda, menggunakan izin yang terkait dengan login konsol Anda.

Di luar konsol dan baris perintah, AWS menawarkan Software Development Kits (SDKs) untuk berbagai bahasa pemrograman, memungkinkan Anda untuk mengintegrasikan fungsionalitas manajemen IAM langsung ke aplikasi Anda. Atau, Anda dapat mengakses IAM secara terprogram menggunakan IAM Query API, yang memungkinkan Anda mengeluarkan permintaan HTTPS langsung ke layanan. Memanfaatkan pendekatan manajemen yang berbeda ini memberi Anda fleksibilitas untuk memasukkan IAM ke dalam alur kerja dan proses yang ada.

Gunakan Konsol Manajemen AWS

AWS Management Console adalah aplikasi web yang terdiri dan mengacu pada koleksi luas konsol layanan untuk mengelola AWS sumber daya. Saat pertama kali masuk, Anda akan melihat halaman beranda konsol. Halaman beranda menyediakan akses ke setiap konsol layanan dan menawarkan satu tempat untuk mengakses informasi untuk melakukan tugas AWS terkait Anda. Layanan dan aplikasi mana yang tersedia untuk Anda setelah masuk ke konsol bergantung pada AWS sumber daya mana yang dapat Anda akses. Anda dapat diberikan izin untuk sumber daya baik melalui asumsi peran, menjadi anggota grup yang telah diberikan izin, atau secara eksplisit diberikan izin. Untuk AWS akun yang berdiri sendiri, pengguna root atau administrator IAM mengonfigurasi akses ke sumber daya. Untuk AWS Organizations, akun manajemen atau administrator yang didelegasikan mengonfigurasi akses ke sumber daya.

Jika Anda berencana untuk memiliki orang yang menggunakan Konsol AWS Manajemen untuk mengelola AWS sumber daya, sebaiknya mengonfigurasi pengguna dengan kredensi sementara sebagai praktik terbaik keamanan. Pengguna IAM yang telah mengambil peran, prinsipal federasi, dan pengguna di IAM Identity Center memiliki kredensi sementara, sedangkan pengguna IAM dan

pengguna root memiliki kredensi jangka panjang. Kredensi pengguna root menyediakan akses penuh ke Akun AWS, sementara pengguna lain memiliki kredenal yang menyediakan akses ke sumber daya yang diberikan oleh kebijakan IAM.

Pengalaman masuk berbeda untuk berbagai jenis Konsol Manajemen AWS pengguna.

- Pengguna IAM dan pengguna root masuk dari URL AWS masuk utama (). https://signin.aws.amazon.com Setelah mereka masuk, mereka memiliki akses ke sumber daya di akun yang telah diberikan izin kepada mereka.

  Untuk masuk sebagai pengguna root, Anda harus memiliki alamat email dan kata sandi pengguna root.

  Untuk masuk sebagai pengguna IAM, Anda harus memiliki Akun AWS nomor atau alias, nama pengguna IAM, dan kata sandi pengguna IAM.

  Kami menyarankan Anda membatasi pengguna IAM di akun Anda untuk situasi tertentu yang memerlukan kredensi jangka panjang, seperti untuk akses darurat, dan bahwa Anda menggunakan pengguna root hanya untuk tugas-tugas yang memerlukan kredensi pengguna root.

  Untuk kenyamanan, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun. Lain kali pengguna membuka halaman mana pun di Konsol Manajemen AWS, konsol menggunakan cookie untuk mengarahkan pengguna ke halaman masuk akun.

  Keluar dari konsol saat Anda menyelesaikan sesi untuk mencegah penggunaan kembali login sebelumnya.
- Pengguna IAM Identity Center masuk menggunakan portal AWS akses khusus yang unik untuk organisasi mereka. Setelah mereka masuk, mereka dapat memilih akun atau aplikasi mana yang akan diakses. Jika mereka memilih untuk mengakses akun, mereka memilih set izin yang ingin mereka gunakan untuk sesi manajemen.
- Prinsipal federasi OIDC dan SALL dikelola di penyedia identitas eksternal yang ditautkan ke Akun AWS login menggunakan portal akses perusahaan khusus. Sumber AWS daya yang tersedia bagi pengguna tergantung pada kebijakan yang dipilih oleh organisasi mereka.

> **Note**
>
> Untuk memberikan tingkat keamanan tambahan, pengguna root, pengguna IAM, dan pengguna di IAM Identity Center dapat memiliki otentikasi multi-faktor (MFA) yang AWS diverifikasi sebelum memberikan akses ke sumber daya. AWS Ketika MFA diaktifkan, Anda juga harus memiliki akses ke perangkat MFA untuk masuk.

Untuk mempelajari lebih lanjut tentang cara pengguna yang berbeda masuk ke konsol manajemen, lihat Masuk ke Konsol AWS Manajemen di Panduan Pengguna AWS Masuk.

AWS Alat Baris Perintah

Anda dapat menggunakan alat baris AWS perintah untuk mengeluarkan perintah di baris perintah sistem Anda untuk melakukan IAM dan AWS tugas. Menggunakan baris perintah dapat lebih cepat dan lebih nyaman dibandingkan konsol. Alat baris perintah juga berguna jika Anda ingin membangun skrip yang melakukan AWS tugas.

AWS menyediakan dua set alat baris perintah: AWS Command Line Interface(AWS CLI) dan AWS Tools for Windows PowerShell. Untuk informasi tentang menginstal dan menggunakan AWS CLI, lihat Panduan AWS Command Line Interface Pengguna. Untuk informasi tentang menginstal dan menggunakan Alat untuk Windows PowerShell, lihat Panduan Alat AWS untuk PowerShell Pengguna.

Setelah masuk ke konsol, Anda dapat menggunakan AWS CloudShell dari browser untuk menjalankan perintah CLI atau SDK. Izin untuk mengakses AWS sumber daya didasarkan pada kredenal yang Anda gunakan untuk masuk ke konsol. Tergantung pada pengalaman Anda, Anda mungkin menemukan CLI menjadi metode yang lebih efisien untuk mengelola Anda. Akun AWS Untuk informasi selengkapnya, lihat Gunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management

AWS Antarmuka Baris Perintah (CLI) dan Kit Pengembangan Perangkat Lunak () SDKs

Pengguna IAM Identity Center dan IAM menggunakan metode yang berbeda untuk mengautentikasi kredensialnya ketika mereka mengautentikasi melalui CLI atau antarmuka aplikasi () yang terkait. APIs SDKs

Kredensi dan pengaturan konfigurasi terletak di beberapa tempat, seperti variabel sistem atau lingkungan pengguna, file AWS konfigurasi lokal, atau secara eksplisit dideklarasikan pada baris perintah sebagai parameter. Lokasi tertentu lebih diutamakan daripada yang lain.

Baik IAM Identity Center dan IAM menyediakan kunci akses yang dapat digunakan dengan CLI atau SDK. Kunci akses Pusat Identitas IAM adalah kredensi sementara yang dapat disegarkan secara otomatis dan direkomendasikan melalui kunci akses jangka panjang yang terkait dengan pengguna IAM.

Untuk mengelola Anda Akun AWS menggunakan CLI atau SDK Anda dapat menggunakan AWS CloudShell dari browser Anda. Jika Anda menggunakan CloudShell untuk menjalankan perintah CLI atau SDK, Anda harus terlebih dahulu masuk ke konsol. Izin untuk mengakses AWS sumber daya didasarkan pada kredenal yang Anda gunakan untuk masuk ke konsol. Tergantung pada pengalaman Anda, Anda mungkin menemukan CLI menjadi metode yang lebih efisien untuk mengelola Anda. Akun AWS

Untuk pengembangan aplikasi, Anda dapat mengunduh CLI atau SDK ke komputer Anda dan masuk dari prompt perintah atau jendela Docker. Dalam skenario ini, Anda mengonfigurasi otentikasi dan mengakses kredensil sebagai bagian dari skrip CLI atau aplikasi SDK. Anda dapat mengonfigurasi akses terprogram ke sumber daya dengan cara yang berbeda, tergantung pada lingkungan dan akses yang tersedia untuk Anda.

- Opsi yang disarankan untuk mengautentikasi kode lokal dengan AWS layanan adalah IAM Identity Center dan IAM Roles Anywhere
- Opsi yang disarankan untuk mengautentikasi kode yang berjalan dalam AWS lingkungan adalah dengan menggunakan peran IAM atau menggunakan kredensil Pusat Identitas IAM.

Saat masuk menggunakan portal AWS akses, Anda bisa mendapatkan kredensi jangka pendek dari halaman awal tempat Anda memilih set izin. Kredensi ini memiliki durasi yang ditentukan dan tidak disegarkan secara otomatis. Jika Anda ingin menggunakan kredensil ini, setelah masuk ke AWS portal, pilih Akun AWS dan kemudian pilih set izin. Pilih Baris perintah atau akses terprogram untuk melihat opsi yang dapat Anda gunakan untuk mengakses AWS sumber daya secara terprogram atau dari CLI. Untuk informasi selengkapnya tentang metode ini, lihat Mendapatkan dan menyegarkan kredensi sementara di Panduan Pengguna Pusat Identitas IAM. Kredensi ini sering digunakan selama pengembangan aplikasi untuk menguji kode dengan cepat.

Sebaiknya gunakan kredensil IAM Identity Center yang secara otomatis menyegarkan saat mengotomatiskan akses ke sumber daya Anda. AWS Jika Anda telah mengonfigurasi pengguna

dan set izin di IAM Identity Center, Anda menggunakan `aws configure sso` perintah untuk menggunakan wizard baris perintah yang akan membantu Anda mengidentifikasi kredensi yang tersedia untuk Anda dan menyimpannya di profil. Untuk informasi selengkapnya tentang mengonfigurasi profil Anda, lihat Mengkonfigurasi profil Anda dengan `aws configure sso` wizard di Panduan Pengguna Antarmuka Baris AWS Perintah untuk Versi 2.

> **Note**
>
> Banyak contoh aplikasi menggunakan kunci akses jangka panjang yang terkait dengan pengguna IAM atau pengguna root. Anda hanya boleh menggunakan kredensil jangka panjang dalam lingkungan kotak pasir sebagai bagian dari latihan pembelajaran. Tinjau alternatif untuk kunci akses jangka panjang dan rencanakan untuk mentransisikan kode Anda untuk menggunakan kredensi alternatif, seperti kredensi Pusat Identitas IAM atau peran IAM, sesegera mungkin. Setelah mentransisikan kode Anda, hapus kunci akses.

Untuk mempelajari lebih lanjut tentang mengonfigurasi CLI, lihat Menginstal atau memperbarui versi terbaru AWS CLI di Panduan Pengguna Antarmuka Baris Perintah untuk Versi 2 dan Otentikasi dan akses kredensional di AWS Panduan Pengguna Antarmuka Baris Perintah AWS

Untuk mempelajari lebih lanjut tentang mengonfigurasi SDK, lihat Autentikasi Pusat Identitas IAM di Panduan Referensi Alat AWS SDKs dan Peran IAM Di Mana Saja di Panduan Referensi Alat dan Alat.AWS SDKs

Gunakan AWS SDKs

AWS menyediakan SDKs (kit pengembangan perangkat lunak) yang terdiri dari pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Python, Ruby, .NET, iOS, Android, dll.). SDKs Menyediakan cara mudah untuk membuat akses terprogram ke IAM dan. AWS Misalnya, SDKs mengurus tugas-tugas seperti menandatangani permintaan secara kriptografis, mengelola kesalahan, dan mencoba ulang permintaan secara otomatis. Untuk informasi tentang AWS SDKs, termasuk cara mengunduh dan menginstalnya, lihat halaman Alat untuk Amazon Web Services.

Gunakan API Kueri IAM

Anda dapat mengakses IAM dan AWS secara terprogram menggunakan IAM Query API, yang memungkinkan Anda mengeluarkan permintaan HTTPS langsung ke layanan. Saat Anda menggunakan Query API, Anda harus menyertakan kode untuk menandatangani permintaan

secara digital menggunakan kredensi Anda. Untuk informasi lebih lanjut, lihat Memanggil API IAM menggunakan permintaan kueri HTTP dan Referensi IAM API.

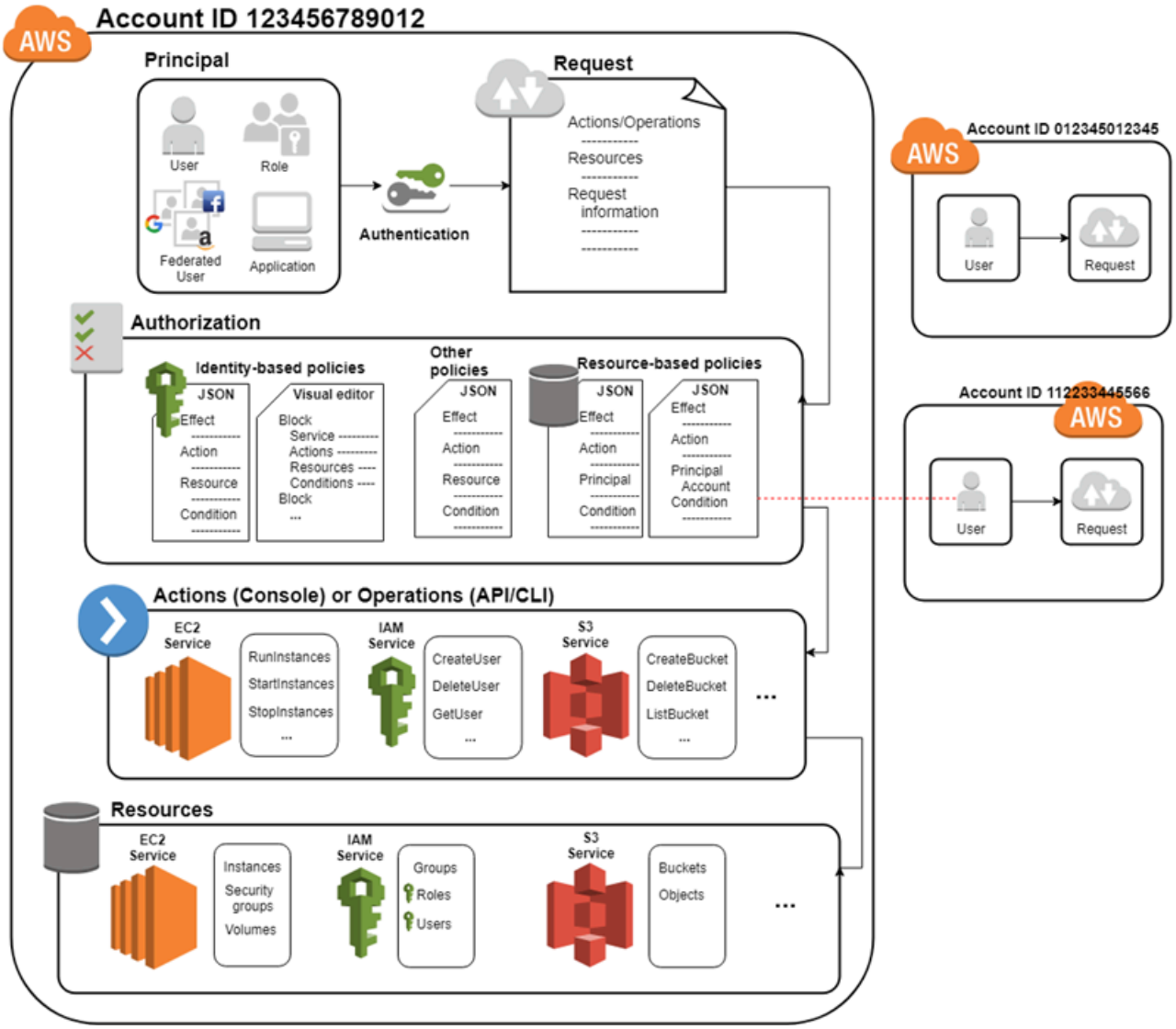
Cara kerja IAM

AWS Identity and Access Management menyediakan infrastruktur yang diperlukan untuk mengontrol otentikasi dan otorisasi untuk Anda Akun AWS.

Pertama, pengguna yang berupa manusia atau aplikasi menggunakan kredensial masuk mereka untuk mengautentikasi AWS. IAM mencocokkan kredentif masuk ke prinsipal (pengguna IAM, prinsipal pengguna AWS STS federasi, peran IAM, atau aplikasi) yang dipercaya oleh Akun AWS dan mengotentikasi prinsipal untuk mengakses. AWS

Selanjutnya, IAM membuat permintaan untuk memberikan akses utama ke sumber daya. IAM memberikan atau menolak akses sebagai tanggapan atas permintaan otorisasi. Misalnya, saat pertama kali masuk ke konsol dan berada di halaman Beranda konsol, Anda tidak mengakses layanan tertentu. Saat memilih layanan, Anda mengirim permintaan otorisasi ke IAM untuk layanan tersebut. IAM memverifikasi bahwa identitas Anda ada dalam daftar pengguna yang berwenang, menentukan kebijakan apa yang mengontrol tingkat akses yang diberikan, dan mengevaluasi kebijakan lain yang mungkin berlaku. Prinsipal di dalam diri Anda Akun AWS atau dari orang lain Akun AWS yang Anda percayai dapat membuat permintaan otorisasi.

Setelah disahkan, kepala sekolah dapat melakukan tindakan atau operasi pada sumber daya di Anda Akun AWS. Misalnya, kepala sekolah dapat meluncurkan Amazon Elastic Compute Cloud instance baru, memodifikasi keanggotaan grup IAM, atau menghapus Amazon Simple Storage Service bucket. Diagram berikut menggambarkan proses ini melalui infrastruktur IAM:



Komponen permintaan

Ketika seorang kepala sekolah mencoba menggunakan Konsol Manajemen AWS, AWS API, atau AWS CLI, kepala sekolah itu mengirimkan permintaan ke AWS. Permintaan tersebut mencakup informasi berikut:

- Tindakan atau operasi — Tindakan atau operasi yang ingin dilakukan oleh kepala sekolah, seperti tindakan di Konsol Manajemen AWS, atau operasi di AWS API AWS CLI or.
- Sumber Daya — AWS Objek sumber daya di mana kepala sekolah meminta untuk melakukan tindakan atau operasi.
- Penanggung jawab – Orang atau aplikasi yang menggunakan entitas (pengguna atau peran) untuk mengirim permintaan. Informasi tentang kepala sekolah termasuk kebijakan izin.

- Data lingkungan — Informasi tentang alamat IP, agen pengguna, status yang diaktifkan SSL, dan stempel waktu.
- Data sumber daya — Data yang terkait dengan sumber daya yang diminta, seperti nama tabel DynamoDB atau tag pada instans Amazon EC2.

AWS mengumpulkan informasi permintaan ke dalam konteks permintaan, yang dievaluasi IAM untuk mengotorisasi permintaan.

Bagaimana prinsipal diautentikasi

Seorang kepala sekolah masuk untuk AWS menggunakan kredensialnya yang diautentikasi IAM untuk mengizinkan kepala sekolah mengirim permintaan. AWS Beberapa layanan, seperti Amazon S3 dan AWS STS, mengizinkan permintaan khusus dari pengguna anonim. Namun, mereka adalah pengecualian dari aturan tersebut. Setiap jenis pengguna melewati otentikasi.

- Pengguna root — Kredentif masuk yang digunakan untuk otentikasi adalah alamat email yang Anda gunakan untuk membuat Akun AWS dan kata sandi yang Anda tentukan pada saat itu.
- Prinsipal federasi — Penyedia identitas Anda mengotentikasi Anda dan meneruskan kreden AWS si Anda ke, Anda tidak perlu masuk langsung. AWS Baik IAM Identity Center dan IAM mendukung federasi identitas.
- Pengguna di Direktori Pusat Identitas AWS IAM (tidak terfederasi) — Pengguna yang dibuat langsung di direktori default Pusat Identitas IAM masuk menggunakan portal AWS akses dan memberikan nama pengguna dan kata sandi Anda.
- Pengguna IAM — Anda masuk dengan memberikan ID akun atau alias, nama pengguna, dan kata sandi Anda. Untuk mengotentikasi beban kerja dari API atau AWS CLI, Anda dapat menggunakan kredenSIAL sementara dengan mengambil peran atau Anda dapat menggunakan kredenSIAL jangka panjang dengan memberikan kunci akses dan kunci rahasia Anda.

  Untuk mempelajari lebih lanjut tentang entitas IAM, lihat Pengguna IAM: danPeran IAM.

AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) dengan semua pengguna untuk meningkatkan keamanan akun Anda. Untuk mempelajari lebih lanjut tentang MFA, lihatAWS Multi-factor otentikasi di IAM.

Dasar-dasar kebijakan otorisasi dan izin

Otorisasi mengacu pada kepala sekolah yang memiliki izin yang diperlukan untuk menyelesaikan permintaan mereka. Selama otorisasi, IAM mengidentifikasi kebijakan yang berlaku untuk permintaan menggunakan nilai dari konteks permintaan. Kemudian menggunakan kebijakan untuk menentukan apakah akan mengizinkan atau menolak permintaan. IAM menyimpan sebagian besar kebijakan izin sebagai dokumen JSON yang menentukan izin untuk entitas utama.

Ada beberapa jenis kebijakan yang dapat memengaruhi permintaan otorisasi. Untuk memberi pengguna izin untuk mengakses AWS sumber daya di akun Anda, Anda dapat menggunakan kebijakan berbasis identitas. Resource-based kebijakan dapat memberikan akses ??? lintas akun. Untuk membuat permintaan di akun lain, kebijakan di akun lain harus mengizinkan Anda mengakses sumber daya dan entitas IAM yang Anda gunakan untuk membuat permintaan harus memiliki kebijakan berbasis identitas yang mengizinkan permintaan tersebut.

IAM memeriksa setiap kebijakan yang berlaku untuk konteks permintaan Anda. Evaluasi kebijakan IAM menggunakan pen olakan eksplisit, yang berarti bahwa jika kebijakan izin tunggal menyertakan tindakan yang ditolak, IAM menolak seluruh permintaan dan berhenti mengevaluasi. Karena permintaan dit olak secara default, kebijakan izin yang berlaku harus mengizinkan setiap bagian dari permintaan Anda untuk IAM untuk mengotorisasi permintaan Anda. Logika evaluasi untuk permintaan dalam satu akun mengikuti aturan dasar ini:

- Secara default, semua permintaan ditolak. (Secara umum, permintaan yang dibuat menggunakan kredensial Pengguna root akun AWS untuk sumber daya di akun selalu diperbolehkan.)
- Izin eksplisit dalam kebijakan izin apa pun (berbasis identitas atau berbasis sumber daya) menggantikan pengaturan bawaan ini.
- Adanya kebijakan kontrol AWS Organizations layanan (SCP) atau kebijakan kontrol sumber daya (RCP), batas izin IAM, atau kebijakan sesi mengesampingkan izin. Jika ada satu atau lebih jenis kebijakan ini, maka semuanya harus mengizinkan permintaan tersebut. Kalau tidak, itu secara implisit ditolak. Untuk informasi selengkapnya tentang SCP dan RCP, lihat Kebijakan otorisasi di AWS Organizations dalam Panduan AWS Organizations Pengguna.
- Penolakan eksplisit dalam kebijakan apa pun mengesampingkan izin apa pun dalam kebijakan apa pun.

Untuk mempelajari selengkapnya, lihat Logika evaluasi kebijakan.

Setelah IAM mengotentikasi dan mengotorisasi prinsipal, IAM menyetujui tindakan atau operasi dalam permintaan mereka dengan mengevaluasi kebijakan izin yang berlaku untuk prinsipal. Setiap AWS layanan mendefinisikan tindakan (operasi) yang mereka dukung, dan menyertakan hal-hal yang dapat Anda lakukan pada sumber daya, seperti melihat, membuat, mengedit, dan menghapus sumber daya tersebut. Kebijakan izin yang berlaku untuk kepala sekolah harus mencakup tindakan yang diperlukan untuk melakukan operasi. Untuk mempelajari selengkapnya tentang cara IAM mengevaluasi kebijakan izin, lihatthe section called “Logika evaluasi kebijakan”.

Layanan mendefinisikan serangkaian tindakan yang dapat dilakukan kepala sekolah pada setiap sumber daya. Saat membuat kebijakan izin, pastikan untuk menyertakan tindakan yang Anda ingin pengguna dapat lakukan. Misalnya, IAM mendukung lebih dari 40 tindakan untuk sumber daya pengguna, termasuk tindakan dasar berikut:

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

Selain itu, Anda dapat menentukan kondisi dalam kebijakan izin yang menyediakan akses ke sumber daya saat permintaan memenuhi ketentuan yang ditentukan. Misalnya, Anda mungkin ingin pernyataan kebijakan berlaku setelah tanggal tertentu atau untuk mengontrol akses saat nilai tertentu muncul di API. Untuk menentukan kondisi, Anda menggunakan `Condition` elemen pernyataan kebijakan.

Setelah IAM menyetujui operasi dalam permintaan, kepala sekolah dapat bekerja dengan sumber daya terkait di akun Anda. Sumber daya adalah objek yang ada di dalam layanan. Contohnya meliputi instans Amazon EC2, pengguna IAM, dan bucket Amazon S3. Jika kepala sekolah membuat permintaan untuk melakukan tindakan pada sumber daya yang tidak termasuk dalam kebijakan izin, layanan menolak permintaan tersebut. Misalnya, jika Anda memiliki izin untuk menghapus peran IAM tetapi meminta untuk menghapus grup IAM, permintaan gagal jika Anda tidak memiliki izin untuk menghapus grup IAM. Untuk mempelajari lebih lanjut tentang tindakan, sumber daya, dan kunci kondisi mana yang didukung oleh berbagai AWS layanan, lihat Tind akan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Membandingkan kredensial dan identitas IAM

Identitas yang dikelola AWS Identity and Access Management adalah pengguna IAM, peran IAM, dan grup IAM. Identitas ini adalah tambahan untuk pengguna root Anda yang AWS dibuat bersama dengan Anda Akun AWS.

Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari, bahkan tugas administratif. Sebagai gantinya, sediakan pengguna tambahan dan berikan mereka izin yang diperlukan untuk melakukan tugas yang diperlukan. Anda dapat menambahkan pengguna dengan menambahkan orang ke direktori Pusat Identitas IAM Anda, menyatukan penyedia identitas eksternal dengan Pusat Identitas IAM atau IAM atau membuat pengguna IAM dengan hak istimewa terendah.

Untuk keamanan tambahan, kami sarankan untuk memusatkan akses root untuk membantu Anda mengamankan kredenSIAL pengguna root dari penggunaan AWS Organizations terkel Akun AWS ola Anda secara terpusat. Mengelola akses root secara terpusat untuk akun anggotamemungkinkan Anda menghapus secara terpusat dan mencegah pemulihan kredensia pengguna root jangka panjang, mencegah akses root yang tidak diinginkan dalam skala besar. Setelah mengaktifkan akses root terpusat, Anda dapat mengasumsikan sesi istimewa untuk melakukan tindakan pada akun anggota.

Setelah mengatur pengguna, Anda dapat memberikan akses ke orang tertentu dan memberi mereka izin untuk mengakses sumber daya. Akun AWS

Sebagai praktik terbaik, Anda AWS menyarankan agar pengguna manusia mengambil peran IAM untuk mengakses AWS sehingga mereka menggunakan kredenSIAL sementara. Jika Anda mengelola identitas di direktori Pusat Identitas IAM atau menggunakan federasi dengan penyedia identitas, Anda mengikuti praktik terbaik.

Istilah

Istilah-istilah ini biasanya digunakan saat bekerja dengan identitas IAM:

Sumber Daya IAM

Layanan IAM menyimpan sumber daya ini. Anda dapat menambahkan, mengedit, dan menghapusnya dari Konsol.

- Pengguna IAM
- Grup IAM

- IAM Role
- Kebijakan izin
- Identity-provider objek

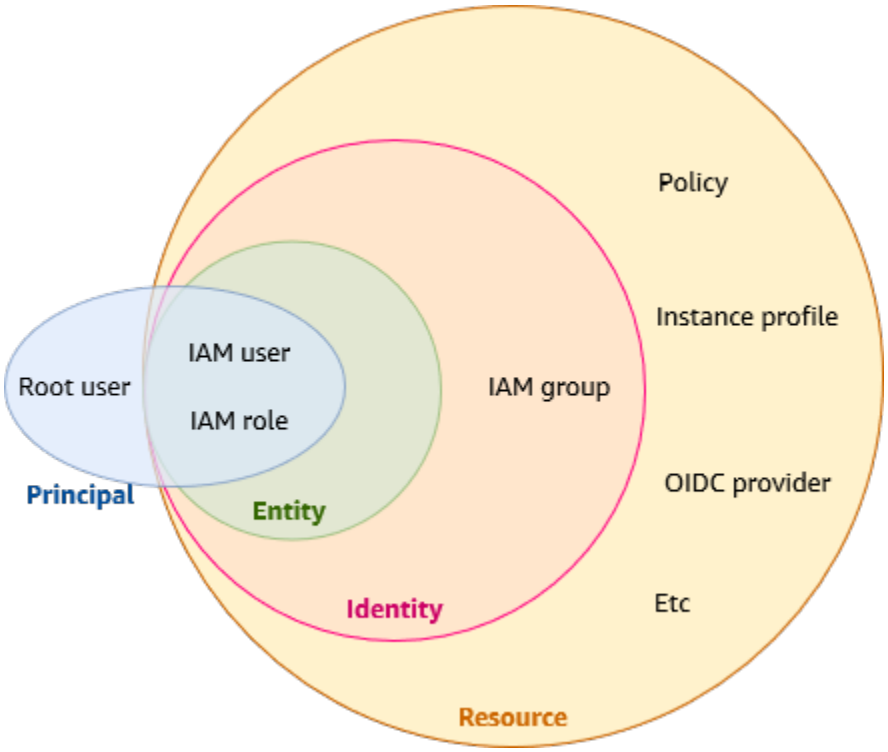
Entitas IAM

Sumber daya IAM yang digunakan untuk AWS otentikasi. Tentukan entitas sebagai Prinsipal dalam kebijakan berbasis sumber daya.

- Pengguna IAM
- IAM Role

Identitas IAM

Sumber daya IAM yang diotorisasi dalam kebijakan untuk melakukan tindakan dan mengakses sumber daya. Identitas termasuk pengguna IAM, grup IAM, dan peran IAM.



Kepala Sekolah

Pengguna Pengguna root akun AWS IAM atau peran IAM yang dapat membuat permintaan untuk tindakan atau operasi pada sumber daya. AWS Prinsip termasuk pengguna manusia, beban kerja, prinsipal federasi, dan peran yang diasumsikan. Setelah otentikasi, IAM memberikan prinsipal kredenSIAL permanen atau sementara untuk membuat permintaan AWS, tergantung pada jenis prinsipal.

Pengguna manusia juga dikenal sebagai identitas manusia, seperti orang, administrator, pengembang, operator, dan konsumen aplikasi Anda.

Beban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi, proses, alat operasional, dan komponen lainnya.

Prinsipal federasi adalah pengguna yang identitas dan kredensialnya dikelola oleh penyedia identitas lain, seperti Active Directory, Okta, atau Microsoft Entra.

Peran IAM adalah identitas IAM yang dapat Anda buat di akun Anda yang memiliki izin khusus yang menentukan apa yang dapat dan tidak dapat dilakukan identitas. Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya.

IAM memberi pengguna IAM dan pengguna root kredenSIAL jangka panjang dan peran IAM kredenSIAL sementara. pengguna di Pusat Identitas AWS IAM, Kepala sekolah federasi OIDC dan SAML mengambil peran IAM saat mereka masuk AWS, yang memberi mereka kredensia sementara. Sebagai praktik terbaik, sebaiknya Anda mewajibkan pengguna manusia dan beban kerja untuk mengakses AWS sumber daya menggunakan kredenSIAL sementara.

Perbedaan antara pengguna IAM dan pengguna di Pusat Identitas IAM

Pengguna IAM bukan akun terpisah; mereka adalah pengguna individu dalam akun Anda. Setiap pengguna memiliki kata sandi mereka sendiri untuk akses ke Konsol Manajemen AWS. Anda juga dapat membuat access key individu untuk setiap pengguna sehingga pengguna dapat membuat permintaan terprogram untuk bekerja dengan sumber daya di akun Anda.

Pengguna IAM dan kunci aksesnya memiliki kredenSIAL jangka panjang ke AWS sumber daya Anda. Penggunaan utama bagi pengguna IAM adalah memberikan beban kerja yang tidak dapat menggunakan peran IAM kemampuan untuk membuat permintaan terprogram ke AWS layanan menggunakan API atau CLI.

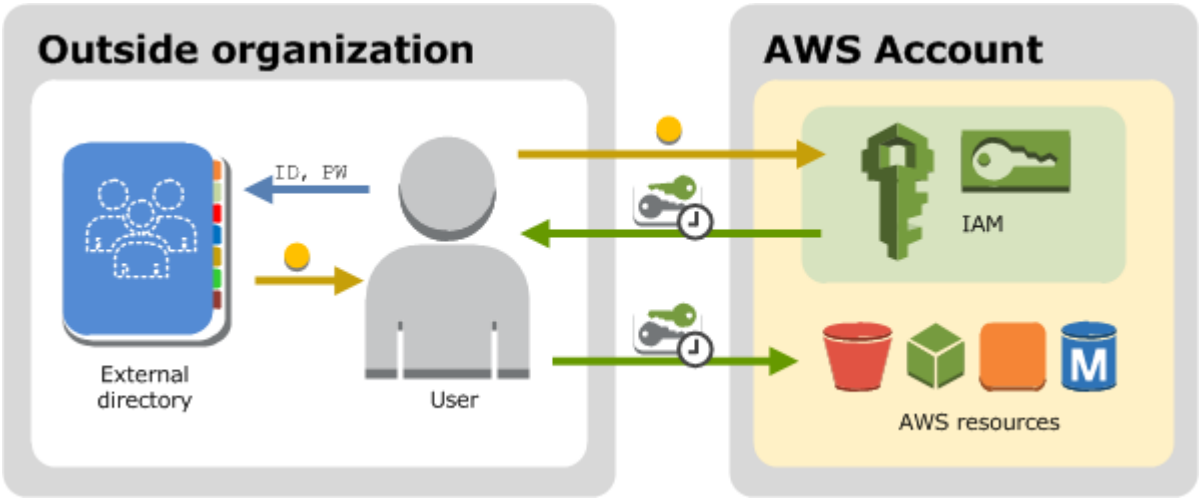
> **Note**
>
> Untuk skenario di mana Anda memerlukan pengguna IAM dengan akses terprogram dan kredenSIAL jangka panjang, sebaiknya perbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Identitas tenaga kerja (orang pengguna di Pusat Identitas AWS IAM) memiliki kebutuhan izin yang berbeda tergantung pada peran yang mereka lakukan dan dapat bekerja dalam berbagai hal Akun AWS di seluruh organisasi. Jika Anda memiliki kasus penggunaan yang memerlukan kunci akses, Anda dapat mendukung kasus penggunaan tersebut dengan pengguna di Pusat Identitas AWS IAM. Orang yang masuk melalui portal AWS akses dapat memperoleh kunci akses dengan kredenSIAL jangka pendek ke AWS sumber daya Anda. Untuk manajemen akses terpusat, sebaiknya gunakan AWS IAM Identity Center (Pusat Identitas IAM) untuk mengelola akses ke akun dan izin dalam akun tersebut. Pusat Identitas IAM secara otomatis dikonfigurasi dengan direktori Pusat Identitas sebagai sumber identitas default tempat Anda dapat menambahkan orang dan grup, dan menetapkan tingkat akses mereka ke AWS sumber daya Anda. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Perbedaan utama antara kedua jenis pengguna ini adalah bahwa pengguna di Pusat Identitas IAM secara otomatis mengambil peran IAM ketika mereka masuk AWS sebelum mengakses konsol manajemen atau sumber daya. AWS Peran IAM memberikan kredenSIAL sementara setiap kali pengguna masuk. AWS Agar pengguna IAM dapat masuk menggunakan peran IAM, mereka harus memiliki izin untuk mengambil dan mengganti peran dan mereka harus secara eksplisit memilih untuk beralih ke peran yang ingin mereka ambil setelah mengakses akun. AWS

Menggabungkan pengguna dari sumber identitas yang ada

Jika pengguna di organisasi Anda sudah diautentikasi saat mereka masuk ke jaringan perusahaan Anda, Anda tidak perlu membuat pengguna atau pengguna IAM terpisah di Pusat Identitas IAM untuk mereka. Sebagai gantinya, Anda dapat menggabungkan identitas pengguna tersebut menjadi AWS menggunakan IAM atau. AWS IAM Identity Center Prinsipal federasi OIDC dan SAML mengambil peran IAM yang memberi mereka izin untuk mengakses sumber daya tertentu. Untuk informasi lebih lanjut tentang peran, lihat Istilah dan konsep peran.



Federasi berguna dalam kasus-kasus ini:

- Pengguna Anda sudah ada di direktori perusahaan.

  Jika direktori perusahaan Anda kompatibel dengan Security Assertion Markup Language 2.0 (SAML 2.0), Anda dapat mengonfigurasi direktori perusahaan untuk menyediakan akses masuk tunggal (SSO) ke pengguna Anda. Konsol Manajemen AWS Untuk informasi selengkapnya, lihat Skenario umum untuk kredensial sementara.

  Jika direktori perusahaan Anda tidak kompatibel dengan SAML 2.0, Anda dapat membuat aplikasi broker identitas untuk menyediakan akses masuk tunggal (SSO) ke Konsol Manajemen AWS untuk pengguna Anda. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

  Jika direktori perusahaan Anda adalah Microsoft Active Directory, Anda dapat menggunakannya AWS IAM Identity Center untuk menghubungkan direktori yang dikelola sendiri di Active Directory atau direktori AWS Directory Service untuk membangun kepercayaan antara direktori perusahaan Anda dan direktori Anda Akun AWS.

  Jika Anda menggunakan penyedia identitas eksternal (IdP) seperti Okta atau Microsoft Entra untuk mengelola pengguna, Anda dapat menggunakannya AWS IAM Identity Center untuk membangun kepercayaan antara IdP Anda dan Anda. Akun AWS Untuk informasi selengkapnya, lihat Menghubun gkan ke penyedia identitas eksternal di Panduan AWS IAM Identity Center Pengguna.

- Pengguna Anda sudah memiliki identitas Internet.

  Jika Anda membuat aplikasi seluler atau aplikasi berbasis web yang memungkinkan pengguna mengidentifikasi diri mereka melalui penyedia identitas Internet seperti Login with Amazon, Facebook, Google, atau penyedia identitas apa pun yang kompatibel dengan OpenID Connect (OIDC), aplikasi dapat menggunakan federasi untuk mengakses AWS. Untuk informasi selengkapnya, lihat Federasi OIDC.

  > **Tip**
  >
  > Untuk menggunakan federasi identitas dengan penyedia identitas Internet, kami sarankan Anda menggunakan Amazon Cognito.

Metode yang berbeda untuk menyediakan akses pengguna

Berikut adalah cara Anda dapat memberikan akses ke AWS sumber daya Anda.

Jenis akses pengguna	Kapan itu digunakan?	Di mana informasi lebih lanjut?
Akses masuk tunggal untuk orang-orang, seperti pengguna tenaga kerja Anda, ke AWS sumber daya menggunakan Pusat Identitas IAM	IAM Identity Center menyediakan tempat sentral yang menyatukan administrasi pengguna dan akses mereka ke Akun AWS dan aplikasi cloud. Anda dapat mengatur penyimpanan identitas dalam IAM Identity Center atau Anda dapat mengonfigurasi federasi dengan penyedia identitas (IdP) yang ada. Praktik terbaik keamanan merekomendasikan pemberian kredensia terbatas kepada AWS sumber daya kepada pengguna manusia Anda. Orang-orang memiliki pengalaman masuk yang lebih mudah dan Anda mempertahankan kontrol atas akses mereka ke sumber daya dari satu sistem. IAM Identity Center mendukung otentikasi multi-faktor (MFA) untuk keamanan akun tambahan.	Untuk informasi selengkapnya tentang menyiapkan Pusat Identitas IAM, lihat Memulai di Panduan AWS IAM Identity Center Pengguna Untuk informasi selengkapnya tentang menggunakan MFA di Pusat Identitas IAM, lihat Multi-factor otentikasi di Panduan Pengguna AWS IAM Identity Center

Jenis akses pengguna	Kapan itu digunakan?	Di mana informasi lebih lanjut?
Akses federasi untuk pengguna manusia, seperti pengguna tenaga kerja Anda, ke AWS layan an yang menggunak an penyedia identitas IAM () IdPs	Dukungan IAM IdPs yang kompatibel dengan OpenID Connect (OIDC) atau SAML 2.0 (Security Assertion Markup Language 2.0). Setelah Anda membuat penyedia identitas IAM, buat satu atau beberapa peran IAM yang dapat ditugaskan secara dinamis ke prinsipal federasi.	Untuk informasi selengkapnya tentang penyedia identitas IAM dan federasi, lihatPenyedia identitas dan federasi ke AWS.
Cross-acc ount akses antara Akun AWS	Anda ingin berbagi akses ke AWS sumber daya tertentu dengan pengguna lain Akun AWS. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, beberapa AWS layanan mendukung kebijakan berbasis sumber daya yang memungkinkan Anda melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy).	Untuk informasi selengkapnya tentang peran IAM, lihatPeran IAM. Untuk mengetahui informasi selengkapnya tentang peran terkait layanan, lihat Buat peran tertaut layanan. Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat. AWS layanan yang bekerja dengan IAM Temukan layanan yang memiliki Ya di kolom Service-Linked Peran. Untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut, pilih tautan yang terkait dengan Ya di kolom itu.

Jenis akses pengguna	Kapan itu digunakan?	Di mana informasi lebih lanjut?
Long-term kredensional untuk pengguna IAM yang ditunjuk di Anda Akun AWS	Anda mungkin memiliki kasus penggunaan tertentu yang memerlukan kredenSIAL jangka panjang dengan pengguna IAM di AWS dalamnya. Anda dapat menggunakan IAM untuk membuat pengguna IAM ini di Anda Akun AWS, dan menggunakan IAM untuk mengelola izin mereka. Beberapa kasus penggunaan meliputi: • Beban kerja yang tidak dapat menggunakan peran IAM • Third-party AWS klien yang memerlukan akses terprogram melalui kunci akses • Service-specific kredenSIAL untuk AWS CodeCommit atau Amazon Keyspaces • AWS IAM Identity Center tidak tersedia untuk akun Anda dan Anda tidak memiliki penyedia identitas lain Sebagai praktik terbaik dalam skenario di mana Anda memerlukan pengguna IAM	Untuk informasi selengkapnya tentang menyiapkan pengguna IAM, lihatBuat pengguna IAM di Akun AWS. Untuk informasi selengkapnya tentang kunci akses pengguna IAM, lihatKelola access key untuk pengguna IAM. Untuk informasi selengkapnya tentang kredenSIAL khusus layanan untuk AWS CodeCommit atau Amazon Keyspaces, lihat dan. Kredensi IAM untuk: Kredensi CodeCommit Git, kunci SSH, dan kunci akses AWS Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Jenis akses pengguna	Kapan itu digunakan?	Di mana informasi lebih lanjut?
	dengan akses ter program dan kredenSIAL jangka panjang, kami sarankan Anda memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.	

Mendukung akses pengguna terprogram

Pengguna memerlukan akses terprogram jika mereka ingin berinteraksi dengan AWS di luar. Konsol Manajemen AWS Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses: AWS

- Jika Anda mengelola identitas di Pusat Identitas IAM, AWS API memerlukan profil, dan AWS Command Line Interface memerlukan profil atau variabel lingkungan.
- Jika Anda memiliki pengguna IAM, AWS API dan kunci akses AWS Command Line Interface memerlukan. Jika memungkinkan, buat kredensial sementara yang terdiri dari ID kunci akses, kunci akses rahasia, dan token keamanan yang menunjukkan masa berlaku kredensial.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Opsi	Informasi selengkapnya
Identitas tenaga kerja (Orang dan pengguna dikelola di Pusat Identitas IAM)	Gunakan kredenSIAL jangka pendek untuk menandata ngani permintaan terprogram ke AWS API AWS CLI atau (secara langsung atau dengan menggunakan AWS SDK).	Untuk itu AWS CLI, ikuti petunjuk dalam Mendap atkan kredenSIAL peran IAM untuk akses CLI di Panduan AWS IAM Identity Center Pengguna. Untuk AWS API, ikuti petunjuk dalam kreden SIAL SSO

Pengguna mana yang membutuhkan akses programatis?	Opsi	Informasi selengkapnya
		di Panduan Refer AWS ensi SDK dan Alat.
Pengguna IAM:	Gunakan kredenSIAL jangka pendek untuk menandata ngani permintaan terprogram ke AWS API AWS CLI atau (secara langsung atau dengan menggunakan AWS SDK).	Ikuti petunjuk dalam Menggunakan kredenSIA L sementara dengan AWS sumber daya.
Pengguna IAM:	Gunakan kredenSIAL jangka panjang untuk menandata ngani permintaan terprogram ke AWS API AWS CLI atau (secara langsung atau dengan menggunakan AWS SDK). (Tidak direkomendasikan)	Ikuti petunjuk dalam M engelola kunci akses untuk pengguna IAM.
Kepala sekolah federasi	Gunakan operasi API AWS STS untuk membuat sesi baru dengan kredentif keamanan sementara yang menyertak an pasangan kunci akses dan token sesi.	Untuk penjelasan tentang operasi API, lihat the section called “Minta kredentif keamanan sementara”

Bagaimana izin dan kebijakan menyediakan manajemen akses

Bagian manajemen akses AWS Identity and Access Management (IAM) membantu Anda menentukan apa yang dapat dilakukan entitas utama dalam akun. Entitas utama adalah orang atau aplikasi yang diautentikasi menggunakan entitas IAM (pengguna IAM atau peran IAM). Manajemen akses sering disebut sebagai otorisasi. Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke identitas IAM (pengguna IAM, grup IAM, atau peran IAM) atau sumber

daya. AWS Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal menggunakan entitas IAM (pengguna IAM atau peran IAM) untuk membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Kebijakan dan akun

Jika Anda mengelola satu akun AWS, maka Anda menentukan izin dalam akun tersebut menggunakan kebijakan. Jika Anda mengelola izin di beberapa akun, akan lebih sulit untuk mengelola izin untuk pengguna IAM Anda. Anda dapat menggunakan peran IAM, kebijakan berbasis sumber daya, atau daftar kontrol akses () ACLs untuk izin lintas akun. Namun, jika Anda memiliki beberapa akun, sebaiknya gunakan AWS Organizations layanan ini untuk membantu Anda mengelola izin tersebut. Untuk informasi lebih lanjut, lihat Apa itu AWS Organizations? dalam AWS Organizations User Guide.

Kebijakan dan pengguna

Pengguna IAM adalah identitas di. Akun AWS Saat Anda membuat pengguna IAM, mereka tidak dapat mengakses apa pun di akun Anda hingga Anda memberi izin kepada mereka. Anda memberikan izin kepada pengguna IAM dengan membuat kebijakan berbasis identitas, yang merupakan kebijakan yang dilampirkan pada pengguna IAM atau grup IAM yang menjadi milik pengguna IAM. Contoh berikut menunjukkan kebijakan JSON yang mengizinkan pengguna IAM untuk melakukan semua tindakan Amazon DynamoDB (`dynamodb:*`) pada `Books` tabel di akun dalam Wilayah. `123456789012` `us-east-2`

JSON

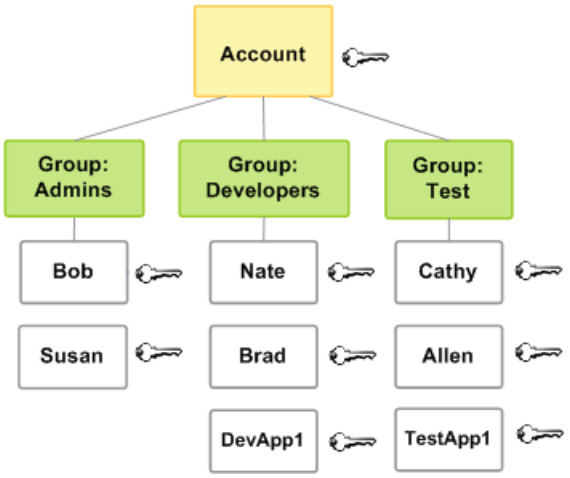
```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Setelah Anda melampirkan kebijakan ini ke pengguna IAM Anda, pengguna memiliki izin untuk melakukan semua tindakan dalam `Books` tabel instance DynamoDB Anda. Sebagian besar pengguna IAM memiliki beberapa kebijakan yang digabungkan untuk mewakili total izin yang diberikan.

Tindakan atau sumber daya yang tidak diizinkan secara eksplisit oleh kebijakan ditolak secara default. Misalnya, jika kebijakan sebelumnya adalah kebijakan tunggal yang dilampirkan ke pengguna, maka pengguna tersebut dapat melakukan tindakan DynamoDB pada `Books` tabel, tetapi tidak dapat melakukan tindakan pada tabel lain. Demikian pula, pengguna tidak diizinkan untuk melakukan tindakan apa pun di Amazon EC2, Amazon S3, atau di layanan AWS lain karena izin untuk bekerja dengan layanan tersebut tidak disertakan dalam kebijakan.

Kebijakan dan grup IAM

Anda dapat mengatur pengguna IAM ke dalam grup IAM dan melampirkan kebijakan ke grup IAM. Dalam hal ini, pengguna IAM individu masih memiliki kredensialnya sendiri, tetapi semua pengguna IAM di grup IAM memiliki izin yang dilampirkan ke grup IAM. Gunakan grup IAM untuk pengelolaan izin yang lebih mudah.



Pengguna IAM atau grup IAM dapat memiliki beberapa kebijakan yang dilampirkan padanya yang memberikan izin berbeda. Dalam hal ini, kombinasi kebijakan menentukan izin efektif untuk prinsipal. Jika prinsipal tidak memiliki `Allow` izin eksplisit untuk tindakan dan sumber daya, prinsipal tidak memiliki izin tersebut.

Sesi dan peran pengguna gabungan

Prinsipal federasi tidak memiliki identitas permanen seperti yang dilakukan pengguna Akun AWS IAM. Untuk menetapkan izin ke prinsipal federasi, Anda dapat membuat entitas yang disebut peran

dan menentukan izin untuk peran tersebut. Saat prinsipal federasi SAFL atau OIDC masuk AWS, pengguna dikaitkan dengan peran tersebut dan diberikan izin yang ditentukan dalam peran tersebut. Untuk informasi selengkapnya, lihat Membuat peran untuk penyedia identitas pihak ketiga.

Kebijakan berbasis identitas dan berbasis sumber daya

Kebijakan berbasis identitas adalah kebijakan perizinan yang Anda lampirkan ke identitas IAM, seperti pengguna, grup, atau peran IAM. Kebijakan berbasis sumber daya adalah kebijakan izin yang Anda lampirkan ke sumber daya seperti bucket Amazon S3 atau kebijakan kepercayaan peran IAM.

Kebijakan berbasis identitas mengendalikan tindakan apa yang dapat dilakukan oleh identitas, pada sumber daya mana, dan dengan kondisi apa. Kebijakan berbasis identitas selanjutnya dapat dikategorikan menjadi:

- Kebijakan terkelola — Kebijakan berbasis identitas mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Anda. Akun AWS Anda dapat menggunakan dua tipe kebijakan terkelola:
  - AWS kebijakan terkelola — Kebijakan terkelola yang dibuat dan dikelola oleh AWS. Jika Anda baru menggunakan kebijakan, kami sarankan Anda memulai dengan menggunakan kebijakan AWS terkelola.
  - Kebijakan terkelola pelanggan — Kebijakan terkelola yang Anda buat dan kelola di Anda Akun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda daripada kebijakan yang AWS dikelola. Anda dapat membuat, mengedit, dan memvalidasi kebijakan IAM di editor visual atau dengan membuat dokumen kebijakan JSON secara langsung. Untuk informasi selengkapnya, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan dan Edit kebijakan IAM.
- Kebijakan inline – Kebijakan yang Anda buat dan kelola dan yang disematkan secara langsung ke dalam satu pengguna, grup, atau peran. Dalam kebanyakan kasus, kami tidak menyarankan penggunaan kebijakan inline.

Kebijakan berbasis sumber daya mengontrol tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dengan kondisi seperti apa. Kebijakan berbasis sumber daya merupakan kebijakan inline, dan tidak ada kebijakan berbasis sumber daya terkelola. Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya.

Layanan IAM mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang Anda lampirkan ke peran IAM. Karena peran IAM adalah identitas dan sumber daya yang mendukung kebijakan berbasis sumber daya, Anda harus melampirkan kebijakan kepercayaan dan kebijakan berbasis identitas ke peran IAM. Kebijakan kepercayaan menentukan entitas utama mana (akun, pengguna, peran, dan prinsip pengguna AWS STS federasi) yang dapat mengambil peran tersebut. Untuk mempelajari bagaimana peran IAM berbeda dengan kebijakan berbasis sumber daya lainnya, lihat Akses sumber daya lintas akun di IAM.

Untuk melihat layanan mana yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari selengkapnya tentang kebijakan berbasis sumber daya, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya.

Tentukan izin berdasarkan atribut dengan otorisasi ABAC

Attribute-based Access Control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. AWS memanggil tag atribut ini. Anda dapat melampirkan tag ke sumber daya IAM, termasuk entitas IAM (pengguna IAM atau peran IAM) dan ke sumber daya. AWS Anda dapat membuat kebijakan ABAC tunggal atau set kecil kebijakan untuk penanggung jawab IAM Anda. Anda dapat mendesain kebijakan ABAC yang mengizinkan operasi saat tag prinsipal cocok dengan tag sumber daya. Sistem atribut ABAC yang menyediakan konteks pengguna tinggi dan kontrol akses granular. Karena ABAC berbasis atribut, ABAC dapat melakukan otorisasi dinamis untuk data atau aplikasi yang memberikan atau mencabut akses secara real time. ABAC membantu dalam lingkungan yang skala dan dalam situasi di mana identitas atau manajemen kebijakan sumber daya telah menjadi kompleks.

Misalnya, Anda dapat membuat tiga peran IAM dengan kunci `access-project` tag. Tetapkan nilai tag dari peran IAM pertama ke`Heart`, yang kedua ke`Star`, dan yang ketiga ke`Lightning`. Anda kemudian dapat menggunakan kebijakan tunggal yang memungkinkan akses ketika peran IAM dan AWS sumber daya memiliki nilai `access-project` tag. Untuk tutorial terperinci yang menunjukkan cara menggunakan ABAC di AWS, lihat Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag. Untuk mempelajari tentang layanan yang mendukung ABAC, lihatAWS layanan yang bekerja dengan IAM.
Diagram ini menggambarkan bahwa tag yang diterapkan pada prinsipal harus cocok dengan tag yang diterapkan ke sumber daya agar pengguna diberikan izin ke sumber daya. Tag diterapkan ke grup IAM, grup sumber daya, pengguna individu, dan sumber daya individu.

Perbandingan ABAC dengan model RBAC tradisional

Model otorisasi tradisional yang digunakan dalam IAM adalah kontrol akses berbasis peran (RBAC). RBAC mendefinisikan izin berdasarkan fungsi pekerjaan seseorang, atau peran, yang berbeda dari peran IAM. IAM mencakup kebijakan terkelola untuk fungsi kerja yang menyelaraskan izin fungsi pekerjaan dalam model RBAC.

Di IAM, Anda menerapkan RBAC dengan membuat kebijakan yang berbeda untuk fungsi pekerjaan yang berbeda. Anda kemudian melampirkan kebijakan ke identitas (pengguna IAM, grup IAM, atau peran IAM). Sebagai praktik terbaik, Anda memberikan izin minimum yang diperlukan untuk fungsi pekerjaan. Ini menghasilkan akses hak istimewa paling sedikit. Setiap kebijakan fungsi pekerjaan mencantumkan sumber daya spesifik yang dapat diakses oleh identitas yang ditetapkan kebijakan tersebut. Kerugian menggunakan model RBAC tradisional adalah ketika Anda atau pengguna menambahkan sumber daya baru ke lingkungan Anda, Anda harus memperbarui kebijakan untuk mengizinkan akses ke sumber daya tersebut.

Misalnya, anggaplah Anda memiliki tiga proyek bernama `Heart`, `Star`, dan `Lightning`, yang dikerjakan karyawan Anda. Anda membuat peran IAM untuk setiap proyek. Anda kemudian melampirkan kebijakan ke setiap peran IAM untuk menentukan sumber daya yang dapat diakses oleh siapa pun yang diizinkan untuk mengambil peran IAM. Jika seorang karyawan berganti pekerjaan dalam perusahaan Anda, Anda menetapkan mereka ke peran IAM yang berbeda. Anda dapat menetapkan orang atau program untuk lebih dari satu peran IAM. Namun, `Star` proyek ini mungkin memerlukan sumber daya tambahan, seperti wadah Amazon EC2 baru. Dalam hal ini, Anda harus memperbarui kebijakan yang dilampirkan ke peran `Star` IAM untuk menentukan sumber daya kontainer baru. Jika tidak, anggota `Star` proyek tidak diizinkan mengakses kontainer baru. Diagram ini menggambarkan bahwa kontrol akses berbasis peran mengharuskan setiap identitas diberi kebijakan berbasis fungsi pekerjaan tertentu untuk mengakses sumber daya yang berbeda.

ABAC memberikan keuntungan berikut dibandingkan model RBAC tradisional:

- Skala izin ABAC dengan inovasi. Administrator idak perlu lagi memperbarui kebijakan yang ada untuk memungkinkan akses ke sumber daya baru. Misalnya, anggap Anda merancang strategi ABAC Anda dan tanda `access-project`. Pengembang menggunakan peran IAM dengan `Heart` tag `access-project` =. Saat orang-orang di proyek `Heart` perlu sumber daya Amazon EC2 tambahan, pengembang dapat menciptakan instans Amazon EC2 baru dengan tag `access-project` = `Heart`. Lalu, siapa pun di proyek `Heart` dapat memulai dan menghentikan instans tersebut karena nilai tanda mereka cocok.

- ABAC membutuhkan lebih sedikit kebijakan. Karena Anda tidak perlu membuat kebijakan yang berbeda untuk fungsi pekerjaan yang berbeda, Anda membuat kebijakan yang lebih sedikit. Kebijakan tersebut lebih mudah dikelola.
- Dengan menggunakan ABAC, tim dapat merespons perubahan dan pertumbuhan secara dinamis. Karena izin untuk sumber daya baru secara otomatis diberikan berdasarkan atribut, Anda tidak perlu menetapkan kebijakan identitas secara manual. Misalnya, jika perusahaan Anda sudah mendukung proyek `Heart` dan `Star` menggunakan ABAC, mudah untuk menambahkan proyek `Lightning` baru. Administrator IAM membuat peran IAM baru dengan tag `access-project =Lightning`. Tidak perlu mengubah kebijakan untuk mendukung proyek baru. Siapa pun yang memiliki izin untuk mengambil peran IAM dapat membuat dan melihat instance yang ditandai dengan =. `access-project` `Lightning` Skenario lain adalah ketika seorang anggota tim pindah dari `Heart` proyek ke `Lightning` proyek. Untuk memberikan akses anggota tim ke `Lightning` proyek, administrator IAM menugaskan mereka ke peran IAM yang berbeda. Tidak perlu mengubah kebijakan izin.
- Izin granular dimungkinkan menggunakan ABAC. Saat Anda membuat kebijakan, memberikan hak istimewa terkecil adalah praktik terbaik. Menggunakan RBAC tradisional, Anda menulis kebijakan yang memungkinkan akses ke sumber daya tertentu. Namun, saat Anda menggunakan ABAC, Anda dapat mengizinkan tindakan pada semua sumber daya jika tag sumber daya cocok dengan tag prinsipal.
- Gunakan atribut karyawan dari direktori perusahaan Anda dengan ABAC. Anda dapat mengonfigurasi penyedia SAFL atau OIDC Anda untuk meneruskan tag sesi ke IAM. Ketika karyawan Anda bergabung AWS, IAM menerapkan atribut mereka ke kepala sekolah yang dihasilkan. Kemudian, Anda dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut.

Untuk tutorial terperinci yang menunjukkan cara menggunakan ABAC di AWS, lihat. Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag

Memulai dengan IAM

AWS Identity and Access Management (IAM) membantu Anda mengontrol akses ke Amazon Web Services (AWS) dan sumber daya akun Anda dengan aman. IAM juga dapat menjaga kredensi login Anda tetap pribadi. Anda tidak secara khusus mendaftar untuk menggunakan IAM. Tidak ada biaya untuk menggunakan IAM.

Bagaimana Anda menggunakan IAM tergantung pada fungsi pekerjaan Anda. Lihat Kapan saya harus menggunakan IAM? panduan khusus peran.

Gunakan IAM untuk memberikan identitas, seperti pengguna dan peran, akses ke sumber daya di akun Anda. Misalnya, Anda dapat menggunakan IAM dengan pengguna yang ada di direktori perusahaan Anda yang Anda kelola secara eksternal AWS atau Anda dapat membuat pengguna AWS menggunakan AWS IAM Identity Center. Identitas federasi mengasumsikan peran IAM yang ditentukan untuk mengakses sumber daya yang mereka butuhkan. Untuk informasi selengkapnya tentang Pusat Identitas IAM, lihat Apa itu Pusat Identitas IAM? dalam Panduan AWS IAM Identity Center Pengguna.

> **Note**
>
> IAM terintegrasi dengan beberapa AWS produk. Untuk daftar layanan yang mendukung IAM, lihat AWS layanan yang bekerja dengan IAM.

Untuk mempelajari cara memulai AWS, membuat pengguna administratif AWS Organizations, dan menggunakan beberapa layanan untuk memecahkan masalah seperti membangun dan meluncurkan proyek pertama Anda, lihat Pusat Sumber Daya Mem ulai.

Untuk latihan langsung dengan tugas IAM umum, lihatTutorial IAM. Tugas termasuk membuat peran untuk akses lintas akun dan menggunakan tag untuk kontrol akses berbasis atribut.

Setelah menyiapkan akun, rencanakan cara orang dan beban kerja mengaksesnya. Untuk informasi selengkapnya, lihat Rencanakan akses ke Anda AWS akun.

Kemudian gunakan Bersiaplah untuk izin hak istimewa terkecil untuk mengikuti prinsip hak istimewa terkecil.

Menyiapkan Akun AWS

Sebelum Anda mulai bekerja dengan IAM, pastikan Anda telah menyelesaikan pengaturan awal AWS lingkungan Anda.

AWS mengirimkan email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan membuka https://aws.amazon.com/ dan memilih Akun Saya.

Ketika Anda mendaftar untuk layanan, Anda membuat Akun AWS menggunakan alamat email dan kata sandi. Itu adalah kreden AWS tif pengguna root Anda. Sebagai praktik terbaik, Anda tidak menggunakan kredentif pengguna root Anda AWS untuk mengakses tugas sehari-hari. Hanya gunakan kredenSIAL pengguna root Anda untuk melakukan tugas yang memerlukan kredenSIAL pengguna root. Juga, jangan bagikan kredensibilitas Anda dengan orang lain. Sebagai gantinya, tambahkan orang ke direktori Anda dan beri mereka akses ke direktori Anda Akun AWS.

Untuk mengamankan Anda Pengguna root akun AWS

1. Masuk ke Konsol Manajemen AWS sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.
2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Berikan akses ke konsol penagihan

Pengguna dan peran IAM dalam sebuah tidak Akun AWS dapat mengakses konsol Penagihan dan Manajemen Biaya secara default. Hal ini berlaku bahkan jika mereka memiliki kebijakan IAM yang memberikan akses ke fitur Penagihan tertentu. Untuk memberikan akses, pengguna Akun AWS root harus terlebih dahulu mengaktifkan akses IAM.

> **Note**
>
> Sebagai praktik terbaik keamanan, kami menyarankan Anda menyediakan akses ke sumber daya Anda melalui federasi identitas dengan AWS IAM Identity Center. Bila

Anda mengaktifkan Pusat Identitas IAM bersamaan dengan AWS Organizations, konsol Penagihan dan Manajemen Biaya diaktifkan secara default dengan penagihan gabungan untuk semua orang Akun AWS di organisasi Anda. Untuk informasi selengkapnya, lihat Mengkonsolidasikan penagihan untuk AWS Organizations di Panduan Pengguna Pen agihan dan Manajemen Biaya.

1. Masuk ke Konsol Manajemen AWS dengan kredensional pengguna root Anda (khususnya, alamat email dan kata sandi yang Anda gunakan untuk membuat AWS akun Anda).
2. Pada bilah navigasi, pilih nama akun Anda, lalu pilih Akun.
3. Gulir ke bawah halaman hingga Anda menemukan bagian Pengguna IAM dan Akses Peran ke Informasi Penagihan, lalu pilih Edit.
4. Pilih kotak centang Aktifkan Akses IAM untuk mengaktifkan akses ke halaman konsol Manajemen Penagihan dan Biaya.
5. Pilih Perbarui.

   Halaman menampilkan pesan Akses IAM user/role ke informasi penagihan diaktifkan.

   Important

   Mengaktifkan akses IAM saja tidak memberikan izin apa pun bagi pengguna atau peran untuk melihat halaman konsol Penagihan dan Manajemen Biaya. Anda juga harus melampirkan kebijakan berbasis identitas yang diperlukan ke peran IAM untuk memberikan akses ke konsol penagihan. Peran menyediakan kredentif sementara yang dapat diasumsikan pengguna saat diperlukan.

6. Gunakan Konsol Manajemen AWS untuk membuat peran yang dapat diasumsikan pengguna untuk mengakses konsol penagihan.
7. Pada halaman Tambah izin untuk peran, tambahkan izin ke daftar dan lihat detail tentang sumber daya Penagihan di Anda Akun AWS.

   Kebijakan ter AWS kelola Penag ihan memberikan izin kepada pengguna untuk melihat dan mengedit konsol Penagihan dan Manajemen Biaya. Ini termasuk melihat penggunaan akun, memodifikasi anggaran, dan metode pembayaran. Untuk contoh kebijakan lainnya yang dapat Anda lampirkan ke peran IAM untuk mengontrol akses ke informasi penagihan akun Anda, lihat Contoh kebijakan AWS penagihan di Panduan Pengguna Penagihan dan Manajemen Biaya.

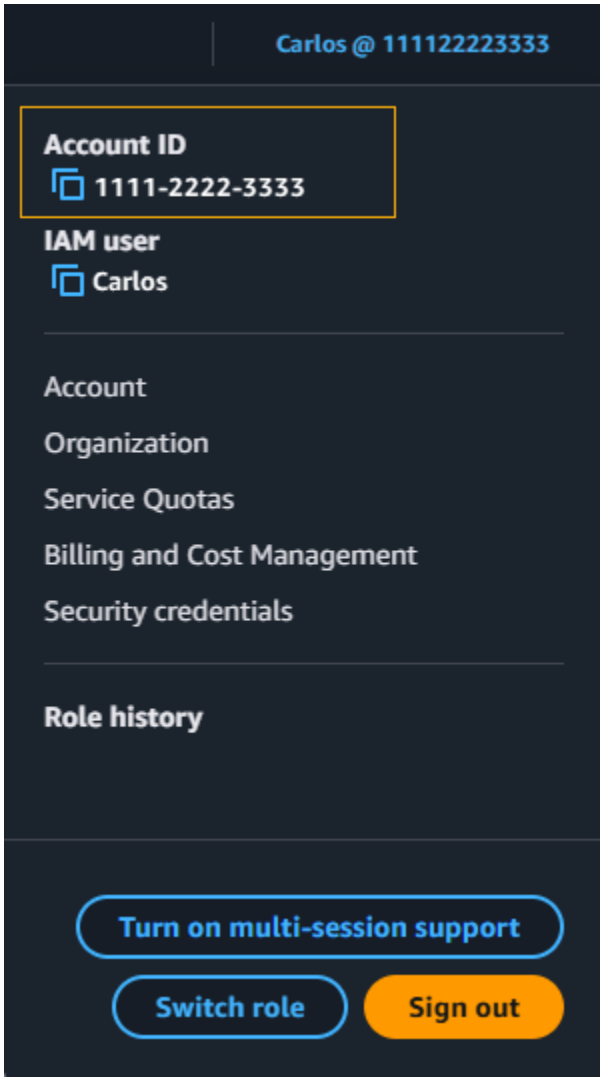
Melihat Anda Akun AWS ID

Jika Anda masuk ke konsol, Anda dapat melihat ID akun untuk Anda Akun AWS menggunakan metode berikut.

Untuk melihat Akun AWS ID

Console

ID AWS akun ditampilkan saat Anda pergi ke Dasbor IAM di Akun AWS bagian tersebut. Anda juga dapat melihat ID akun Anda di bilah navigasi di kanan atas. Pilih nama pengguna Anda, dan ID akun ditampilkan di atas nama pengguna Anda.



AWS CLI

Gunakan perintah berikut untuk melihat ID pengguna, ID akun, dan ARN pengguna Anda:

- aws sts mendapatkan identitas-penelepon

API

Gunakan API berikut untuk melihat ID pengguna, ID akun, dan ARN pengguna Anda:

- GetCallerIdentity

Menggunakan alias untuk Anda Akun AWS ID

ID akun Anda adalah nomor 12 digit yang secara unik mengidentifikasi akun Anda. Secara default, pengguna IAM di akun masuk menggunakan URL web yang menyertakan ID akun. Jika mereka tidak memiliki URL, mereka dapat memberikan ID akun di halaman AWS masuk saat mereka masuk.

URL halaman masuk Anda memiliki format sebagai berikut, secara default.

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

Banyak orang menganggap kata-kata lebih mudah diingat daripada angka, jadi membuat alias untuk ID akun Anda dapat membantu pengguna IAM Anda masuk lebih mudah.

Jika Anda membuat Akun AWS alias untuk Akun AWS ID Anda, URL halaman masuk Anda terlihat seperti contoh berikut.

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

Pertimbangan sebelum membuat alias akun

- Anda hanya Akun AWS dapat memiliki satu alias. Jika Anda membuat alias baru untuk AWS akun Anda, alias baru akan menggantikan alias sebelumnya, dan URL yang berisi alias sebelumnya berhenti berfungsi.
- Alias akun hanya harus berisi digit, huruf kecil, dan tanda hubung. Untuk informasi selengkapnya tentang batasan pada entitas AWS akun, lihatIAM dan AWS STS kuota.
- Alias akun harus unik di semua produk Amazon Web Services dalam partisi jaringan tertentu.

  Partisi adalah grup Wilayah AWS . Setiap AWS akun dicakup ke satu partisi.

  Berikut ini adalah partisi yang didukung:

- `aws`- Wil AWS ayah
- `aws-cn` - Wilayah Tiongkok
- `aws-us-gov` - AWS GovCloud (US) Wilayah

> **Note**
>
> Alias akun bukan rahasia, dan akan muncul di URL halaman masuk publik Anda. Jangan sertakan informasi sensitif apa pun dalam alias akun Anda.
> URL asli yang berisi Akun AWS ID Anda tetap aktif dan dapat digunakan setelah Anda membuat Akun AWS alias.

Untuk petunjuk tentang cara mengatur atau menghapus alias, lihat Membuat alias akun dan Menghapus alias akun.

Membuat alias akun

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:

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

Untuk membuat Akun AWS alias

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Dasbor.
3. Di bagian AWS Akun, di samping Alias Akun, pilih Buat. Jika alias sudah ada, maka pilih Edit.
4. Di kotak dialog, masukkan nama yang ingin Anda gunakan untuk alias Anda, lalu pilih Simpan perubahan.

AWS CLI

Jalankan perintah berikut:

- `aws iam create-account-alias`

API

Untuk membuat alias untuk URL halaman Konsol Manajemen AWS masuk Anda, panggil operasi berikut:

- `CreateAccountAlias`

Menghapus alias akun

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:

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

Untuk menghapus alias akun

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Dasbor.
3. Di bagian AWS Akun, di samping Alias Akun, pilih Hapus.

AWS CLI

Untuk menghapus alias Akun AWS ID, jalankan perintah berikut:

- `aws iam delete-account-alias`

Untuk mengonfirmasi bahwa alias akun dihapus, coba tampilkan alias Akun AWS ID Anda, dengan menjalankan perintah berikut:

- `aws iam list-account-aliases`

API

Untuk menghapus alias Akun AWS ID, panggil operasi berikut:

- `DeleteAccountAlias`

Untuk mengonfirmasi bahwa alias akun dihapus mencoba menampilkan alias Akun AWS ID Anda, dengan memanggil operasi berikut:

- `ListAccountAliases`

> **Note**
>
> Setelah menghapus alias akun Anda, satu-satunya URL masuk untuk akun Anda didasarkan pada ID akun Anda. Setiap upaya untuk terhubung ke URL alias akan gagal dan tidak dialihkan.

Rencanakan akses ke Anda AWS akun

Saat menyiapkan AWS, rencanakan bagaimana Anda ingin orang mengakses AWS akun dan sumber daya Anda. Solusi manajemen identitas yang dirancang dengan baik menjaga akun Anda tetap aman.

Sumber identitas

Menurut praktik terbaik IAM, kami menyarankan agar pengguna manusia dan beban kerja menggunakan kredentif sementara untuk mengakses sumber daya Anda AWS . Anda memberikan kredenSIAL sementara melalui peran IAM. Kedua pengguna yang disatukan ke dalam IAM dan pengguna di Pusat Identitas IAM (baik yang terfederasi atau dibuat di direktori Pusat Identitas IAM) menggunakan peran IAM untuk mengakses sumber daya.

Sebelum mulai menggunakan AWS, rencanakan cara mengatur identitas Anda dengan menggunakan salah satu pendekatan berikut:

- Mengaktifkan Pusat Identitas IAM dengan AWS Organizations dan menambahkan pengguna di Pusat Identitas IAM langsung ke direktori organisasi.

Untuk petunjuk tentang menambahkan pengguna secara langsung ke direktori organisasi Pusat Identitas IAM, lihat Men ambahkan pengguna

- Menggabungkan penyedia identitas eksternal Anda yang ada dengan IAM Identity Center atau IAM.

  Untuk petunjuk tentang menyatukan penyedia identitas eksternal ke direktori organisasi Pusat Identitas IAM, lihat tutorial Memulai yang sesuai.

Manajemen akses

Identifikasi AWS sumber daya dan layanan yang dibutuhkan pengguna Anda. Kemudian tentukan izin akses dan kebijakan untuk setiap pengguna, grup, atau peran.

- Jika Anda menggunakan Pusat Identitas IAM, penyedia identitas IAM, peran IAM, dan kebijakan izin dibuat di setiap AWS akun di organisasi Anda. Peran dan izin ini cocok dengan apa yang Anda tentukan saat menetapkan orang atau grup ke aplikasi atau AWS akun.

  Untuk informasi selengkapnya, lihat Men etapkan akses pengguna dan Mengatur akses masuk tunggal ke aplikasi Anda.
- Anda dapat menggunakan manajer akses akun — fitur IAM yang memungkinkan Anda menetapkan peran IAM yang ada yang Anda buat dan kelola kepada pengguna dan grup Pusat Identitas IAM. Manajer akses akun memberi Anda akses ke set fitur peran IAM lengkap. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.
- Jika Anda menyatukan penyedia identitas Anda secara langsung dengan IAM di dalam Akun AWS, Anda harus membuat peran untuk diasumsikan oleh pengguna Anda. Peran memerlukan kebijakan kepercayaan dan kebijakan izin. Kebijakan kepercayaan menyatakan siapa yang dapat mengambil peran. Kebijakan izin menyatakan AWS tindakan dan sumber daya mana peran diizinkan atau ditolak aksesnya.

  Untuk informasi selengkapnya, lihat Penyedia identitas dan federasi ke AWS

Untuk petunjuk tentang menambahkan lapisan keamanan tambahan untuk masuk, lihatGunakan otentikasi multi-faktor dengan identitas Anda.

Kasus penggunaan untuk pengguna IAM

Pengguna IAM yang Anda buat di Anda Akun AWS memiliki kredenSIAL jangka panjang yang Anda kelola secara langsung.

Dalam hal mengelola akses AWS, pengguna IAM umumnya bukan pilihan terbaik. Ada beberapa alasan utama mengapa Anda harus menghindari mengandalkan pengguna IAM untuk sebagian besar kasus penggunaan Anda.

Pertama, pengguna IAM dirancang untuk akun individual, sehingga mereka tidak berskala dengan baik seiring pertumbuhan organisasi Anda. Mengelola izin dan keamanan untuk sejumlah besar pengguna IAM dapat dengan cepat menjadi tantangan.

Pengguna IAM juga tidak memiliki visibilitas terpusat dan kemampuan audit yang Anda dapatkan dengan solusi manajemen AWS identitas lainnya. Hal ini dapat membuatnya lebih menantang untuk menjaga keamanan dan kepatuhan terhadap peraturan.

Terakhir, menerapkan praktik terbaik keamanan seperti otentikasi multi-faktor, kebijakan kata sandi, dan pemisahan peran jauh lebih mudah dengan pendekatan manajemen identitas yang lebih skalabel.

Alih-alih mengandalkan pengguna IAM, sebaiknya gunakan solusi yang lebih kuat seperti Pusat Identitas IAM dengan AWS Organizations, atau identitas federasi dari penyedia eksternal. Opsi ini akan memberi Anda kontrol, keamanan, dan efisiensi operasional yang lebih baik seiring pertumbuhan AWS lingkungan Anda.

Oleh karena itu, kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan yang tidak didukung oleh pengguna federasi.

Daftar berikut mengidentifikasi kasus penggunaan tertentu yang memerlukan kredenSIAL jangka panjang dengan pengguna IAM di AWS dalamnya. Anda dapat menggunakan IAM untuk membuat pengguna IAM ini di bawah payung AWS akun Anda, dan menggunakan IAM untuk mengelola izin mereka.

- Akses darurat ke AWS akun Anda
- Beban kerja yang tidak dapat menggunakan peran IAM
  - AWS CodeCommit akses
  - Akses Ruang Kunci Amazon (untuk Apache Cassandra)
- Third-party AWS klien

- AWS IAM Identity Center tidak tersedia untuk akun Anda dan Anda tidak memiliki penyedia identitas lain

Buat pengguna IAM untuk akses darurat

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi.

Memiliki pengguna IAM untuk akses darurat adalah salah satu alasan yang disarankan untuk membuat pengguna IAM sehingga Anda dapat mengakses Akun AWS jika penyedia identitas Anda tidak dapat diakses.

> **Note**
>
> Sebagai praktik terbaik keamanan, sebaiknya berikan akses ke sumber daya melalui federasi identitas alih-alih membuat pengguna IAM. Untuk informasi tentang situasi tertentu di mana pengguna IAM diperlukan, lihat Kapan membuat pengguna IAM (bukan peran).

Untuk membuat pengguna IAM untuk akses darurat

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - `access-analyzer:ValidatePolicy`
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateGroup`
> - `iam:CreateLoginProfile`
> - `iam:CreateUser`
> - `iam:GetAccountPasswordPolicy`
> - `iam:GetLoginProfile`
> - `iam:GetUser`
> - `iam:ListAttachedGroupPolicies`

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

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Cari IAM.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih Buat pengguna.

> **Note**
>
> Jika Anda mengaktifkan Pusat Identitas IAM, akan Konsol Manajemen AWS menampilkan pengingat bahwa yang terbaik adalah mengelola akses pengguna di Pusat Identitas IAM. Dalam prosedur ini, pengguna IAM yang Anda buat khusus digunakan hanya jika penyedia identitas Anda tidak tersedia.

4. Pada halaman Tentukan detail pengguna, di bawah Detail pengguna, di Nama pengguna, masukkan nama untuk pengguna baru. Ini adalah nama masuk mereka untuk AWS. Untuk contoh ini, masukkan**`EmergencyAccess`**.

> **Note**
>
> Nama pengguna dapat berupa kombinasi hingga 64 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), tanda at (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Hal ini tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat dua pengguna yang diberi nama TESTUSER dan testuser. Ketika nama pengguna digunakan dalam kebijakan atau sebagai bagian dari ARN, nama tersebut peka huruf besar/kecil. Ketika nama

pengguna terlihat oleh pelanggan di konsol, seperti selama proses masuk, nama pengguna tersebut tidak dibedakan berdasarkan huruf besar-kecilnya.

5. Pilih kotak centang di samping Men yediakan akses pengguna ke Konsol Manajemen AWS — opsional , lalu pilih Saya ingin membuat pengguna IAM.
6. Di bawah Kata sandi konsol, pilih Kata sandi yang dibuat secara otomatis.
7. Kosongkan kotak centang di samping Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan). Karena pengguna IAM ini untuk akses darurat, administrator tepercaya menyimpan kata sandi dan hanya menyediakannya bila diperlukan.
8. Pada halaman Tetapkan izin, di bawah opsi Izin, pilih Tambahkan pengguna ke grup. Kemudian, di bawah Grup pengguna, pilih Buat grup.
9. Pada halaman Buat grup pengguna, di Nama grup pengguna, masukkan**`EmergencyAccessGroup`**. Kemudian, di bawah Kebijakan Iz in, pilih AdministratorAccess.
10. Pilih Buat grup pengguna untuk kembali ke halaman Tetapkan izin.
11. Di bawah Grup pengguna, pilih nama yang **`EmergencyAccessGroup`** Anda buat sebelumnya.
12. Pilih Ber ikutnya untuk melanjutkan ke halaman Review dan buat.
13. Pada halaman T injau dan buat, tinjau daftar keanggotaan grup pengguna yang akan ditambahkan ke pengguna baru. Ketika Anda siap untuk melanjutkan, pilih Buat pengguna.
14. Pada halaman Ambil kata sandi, pilih Unduh file.csv untuk menyimpan file.csv dengan informasi kredensi pengguna (URL koneksi, nama pengguna, dan kata sandi).
15. Simpan file ini untuk digunakan jika Anda perlu masuk ke IAM dan tidak memiliki akses ke penyedia identitas Anda.

Pengguna IAM baru ditampilkan dalam daftar Pengguna. Pilih tautan Nama pengguna untuk melihat detail pengguna.

AWS CLI

1. Buat pengguna bernama**`EmergencyAccess`**.

   - aws iam buat pengguna

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

2. (Opsional) Berikan akses pengguna ke Konsol Manajemen AWS. Ini memerlukan kata sandi. Untuk membuat kata sandi untuk pengguna IAM, Anda dapat menggunakan `--cli-input-json` parameter untuk meneruskan file JSON yang berisi kata sandi.Anda juga harus memberikan pengguna URL halaman masuk akun Anda.

   - buat-profil-masuk aws iam

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

   - Buka `create-login-profile.json` file di editor teks dan masukkan kata sandi yang sesuai dengan kebijakan kata sandi Anda, lalu simpan file tersebut. Contoh:

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

   - Gunakan `aws iam create-login-profile` perintah lagi, lewati `--cli-input-json` parameter untuk menentukan file JSON Anda.

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

   > **Note**
   >
   > Jika kata sandi yang Anda berikan dalam file JSON melanggar kebijakan kata sandi akun Anda, Anda akan menerima `PasswordPolicyViolation` kesalahan. Jika ini terjadi, tinj au kebijakan kata sandi untuk akun Anda dan perbarui kata sandi dalam file JSON agar sesuai dengan persyaratan.

3. Buat**`EmergencyAccessGroup`**, lampirkan AWS kebijakan ter `AdministratorAccess` kelola ke grup, dan tambahkan **`EmergencyAccess`** pengguna ke grup.

> **Note**
>
> Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Setiap kebijakan memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi selengkapnya tentang ARN, lihat ARN IAM Untuk daftar kebijakan terkel AWS ola untuk Layanan AWS, lihat kebijakan ter AWS kelola.

- aws iam buat grup

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

- kebijakan grup lampiran aws iam

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

- aws iam tambahkan pengguna-ke-grup

```
aws iam add-user-to-group \
   --user-name EmergencyAccess \
   --group-name EmergencyAccessGroup
```

- Jalankan perintah aws iam get-group untuk membuat daftar **`EmergencyAccessGroup`** dan anggotanya.

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

Membuat pengguna IAM untuk beban kerja yang tidak dapat menggunakan peran IAM

> **Important**
>
> Sebagai praktik terbaik, kami sarankan Anda meminta pengguna manusia untuk menggunakan kreden SIAL sementara saat mengakses AWS.
> Atau, Anda dapat mengelola identitas pengguna Anda, termasuk pengguna administratif Anda, dengan AWS IAM Identity Center. Sebaiknya gunakan Pusat Identitas IAM untuk mengelola akses ke akun dan izin dalam akun tersebut. Jika Anda menggunakan penyedia identitas eksternal, Anda juga dapat mengonfigurasi izin akses untuk identitas pengguna di Pusat Identitas IAM.

Jika kasus penggunaan Anda memerlukan pengguna IAM dengan akses terprogram dan kredenSIAL jangka panjang, sebaiknya Anda menetapkan prosedur untuk memperbarui kunci akses bila diperlukan. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Untuk melakukan beberapa tugas manajemen akun dan layanan, Anda harus masuk menggunakan kredenSIAL pengguna root. Untuk melihat tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.

Untuk membuat pengguna IAM untuk beban kerja yang tidak dapat menggunakan peran IAM

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateAccessKey`
> - `iam:CreateGroup`
> - `iam:CreateServiceSpecificCredential`
> - `iam:CreateUser`
> - `iam:GetAccessKeyLastUsed`
> - `iam:GetAccountPasswordPolicy`

- iam:GetAccountSummary
- iam:GetGroup
- iam:GetLoginProfile
- iam:GetPolicy
- iam:GetRole
- iam:GetUser
- iam:ListAccessKeys
- iam:ListAttachedGroupPolicies
- iam:ListAttachedUserPolicies
- iam:ListGroupPolicies
- iam:ListGroups
- iam:ListGroupsForUser
- iam:ListInstanceProfilesForRole
- iam:ListMFADevices
- iam:ListPolicies
- iam:ListRoles
- iam:ListRoleTags
- iam:ListSSHPublicKeys
- iam:ListServiceSpecificCredentials
- iam:ListSigningCertificates
- iam:ListUserPolicies
- iam:ListUserTags
- iam:ListUsers
- iam:UploadSSHPublicKey
- iam:UploadSigningCertificate

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.

2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Cari IAM.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih Buat pengguna.
4. Pada halaman Tentukan detail pengguna, lakukan hal berikut:
   a. Untuk Nama pengguna, ketik ***`WorkloadName`***. Ganti ***`WorkloadName`*** dengan nama beban kerja yang akan menggunakan akun.
   b. Pilih Berikutnya.
5. (Opsional) Pada halaman Setel Izin, lakukan hal berikut:
   a. Pilih Tambahkan pengguna ke grup.
   b. Pilih Buat kelompok.
   c. Dalam kotak dialog Buat grup pengguna, untuk Nama grup pengguna ke tik nama yang mewakili penggunaan beban kerja dalam grup. Untuk contoh ini, gunakan nama**`Automation`**.
   d. Di bawah Kebijakan izin pilih kotak centang untuk kebijakan terkel PowerUserAccess ola.

      > **Tip**
      >
      > Masukkan Power ke kotak pencarian kebijakan izin untuk menemukan kebijakan terkelola dengan cepat.

   e. Pilih Buat grup pengguna.
   f. Kembali ke halaman dengan daftar grup IAM, pilih kotak centang untuk grup pengguna baru Anda. Pilih Refresh (Segarkan) jika Anda tidak melihat grup pengguna baru di dalam daftar.
   g. Pilih Berikutnya.
6. (Opsional) Di bagian Tag, tambahkan metadata ke pengguna dengan melampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.
7. Verifikasi keanggotaan grup pengguna untuk pengguna baru. Saat Anda siap untuk melanjutkan, pilih Create user (Buat pengguna).
8. Pemberitahuan status muncul yang memberi tahu Anda bahwa pengguna berhasil dibuat. Pilih Lihat pengguna untuk membuka halaman detail pengguna

9. Pilih tab Kredensibilitas keamanan. Kemudian buat kredenSIAL yang dibutuhkan untuk beban kerja.

   - Kunci akses —Pilih Buat kunci akses untuk menghasilkan dan mengunduh kunci akses untuk pengguna.

     > **Important**
     >
     > Ini adalah satu-satunya kesempatan Anda untuk melihat atau mengunduh kunci akses rahasia, dan Anda harus memberikan informasi ini kepada pengguna Anda sebelum mereka dapat menggunakan AWS API. Simpan access key ID baru pengguna dan secret access key di tempat yang aman dan terlindungi. Anda tidak akan memiliki akses ke kunci rahasia lagi setelah langkah ini.

   - Kunci publik SSH untuk AWS CodeCommit —Pilih Unggah kunci publik SSH untuk mengunggah kunci publik SSH sehingga pengguna dapat berkomunikasi dengan CodeCommit repositori melalui SSH.
   - KredenSIAL HTTPS Git untuk AWS CodeCommit —Pilih Hasilkan kredenSIAL untuk menghasilkan kumpulan kredenSIAL pengguna yang unik untuk digunakan dengan repositori Git. Pilih Unduh kreden sional untuk menyimpan nama pengguna dan kata sandi ke file.csv. Ini adalah satu-satunya waktu informasi tersedia. Jika Anda lupa atau kehilangan kata sandi, Anda harus mengatur ulang kata sandi.
   - KredenSIAL untuk Amazon Keyspaces (untuk Apache Cassandra) —Pilih Hasilkan kreden SIAL untuk menghasilkan kredenSIAL pengguna khusus layanan untuk digunakan dengan Amazon Keyspaces. Pilih Unduh kreden sional untuk menyimpan nama pengguna dan kata sandi ke file.csv. Ini adalah satu-satunya waktu informasi tersedia. Jika Anda lupa atau kehilangan kata sandi, Anda harus mengatur ulang kata sandi.

     > **Important**
     >
     > Service-specific kredenSIAL adalah kredenSIAL jangka panjang yang terkait dengan pengguna IAM tertentu dan hanya dapat digunakan untuk layanan yang dibuat untuknya. Untuk memberikan izin peran IAM atau identitas federasi untuk mengakses semua AWS sumber daya Anda menggunakan kredenSIAL sementara, gunakan AWS otentikasi dengan plugin otentikasi SIGv4 untuk Amazon Keyspaces. Untuk informasi selengkapnya, lihat Menggunakan kredenSIAL sementara untuk terhubung ke Amazon Keyspaces (untuk Apache Cassandra)

menggunakan peran IAM dan plugin SIGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

- X.509 Menandatangani sertifikat —Pilih Buat X.509 Sertifikat jika Anda perlu membuat SOAP-protocol permintaan aman dan berada di Wilayah yang tidak didukung oleh AWS Certificate Manager. ACM adalah alat pilihan untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Untuk informasi selengkapnya tentang menggunakan ACM, lihat Panduan AWS Certificate Manager Pengguna.

Anda telah membuat pengguna dengan akses terprogram dan mengonfigurasinya dengan fungsi PowerUserAccess pekerjaan. Kebijakan izin pengguna ini memberikan akses penuh ke setiap layanan kecuali untuk IAM dan AWS Organizations.

Anda dapat menggunakan proses yang sama ini untuk memberikan beban kerja tambahan akses terprogram ke Akun AWS sumber daya Anda, jika beban kerja tidak dapat mengambil peran IAM. Prosedur ini menggunakan kebijakan ter PowerUserAccess kelola untuk menetapkan izin. Untuk mengikuti praktik terbaik hak istimewa terkecil, pertimbangkan untuk menggunakan kebijakan yang lebih ketat atau membuat kebijakan khusus yang membatasi akses hanya ke sumber daya yang diperlukan oleh program. Untuk mempelajari tentang menggunakan kebijakan yang membatasi izin pengguna ke AWS sumber daya tertentu, lihat Manajemen akses untuk AWS sumber daya danContoh kebijakan berbasis identitas IAM. Untuk menambahkan pengguna tambahan ke grup pengguna setelah dibuat, lihat Mengedit pengguna dalam grup IAM.

AWS CLI

1. Buat pengguna bernama**Automation**.

   - aws iam buat pengguna

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

2. Buat grup pengguna IAM bernama**AutomationGroup**, lampirkan AWS kebijakan ter `PowerUserAccess` kelola ke grup, lalu tambahkan **Automation** pengguna ke grup.

> **Note**
>
> Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Setiap kebijakan memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi selengkapnya tentang ARN, lihat ARN IAM Untuk daftar kebijakan terkel AWS ola untuk Layanan AWS, lihat kebijakan ter AWS kelola.

- aws iam buat grup

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

- kebijakan grup lampiran aws iam

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

- aws iam tambahkan pengguna-ke-grup

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

- Jalankan perintah aws iam get-group untuk membuat daftar **AutomationGroup** dan anggotanya.

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

3. Buat kredentif keamanan yang diperlukan untuk beban kerja.

- Buat kunci akses untuk pengujian - aws iam create-access-key

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

Output dari perintah ini menampilkan kunci akses rahasia dan ID kunci akses. Rekam dan simpan informasi ini di lokasi yang aman. Jika kredenSIAL ini hilang, mereka tidak dapat dipulihkan, dan Anda harus membuat kunci akses baru.

> **Important**
>
> Kunci akses pengguna IAM ini adalah kredenSIAL jangka panjang yang menghadirkan risiko keamanan pada akun Anda. Setelah Anda menyelesaikan pengujian, kami sarankan Anda menghapus kunci akses ini. Jika Anda memiliki skenario di mana Anda mempertimbangkan kunci akses, selidiki apakah Anda dapat mengaktifkan MFA untuk pengguna IAM beban kerja Anda dan menggunakan aws sts get-session-token untuk mendapatkan kredenSIAL sementara untuk sesi alih-alih menggunakan kunci akses IAM.

- Unggah kunci publik SSH untuk AWS CodeCommit - aws iam upload-ssh-public-key

Contoh berikut mengasumsikan bahwa Anda memiliki kunci publik SSH Anda yang disimpan dalam file`sshkey.pub`.

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

- Unggah sertifikat X.509 penandatanganan - aws iam upload-signning-certificate

Unggah X.509 sertifikat jika Anda perlu membuat SOAP-protocol permintaan aman dan berada di Wilayah yang tidak didukung oleh AWS Certificate Manager. ACM adalah alat pilihan untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Untuk informasi selengkapnya tentang menggunakan ACM, lihat Panduan AWS Certificate Manager Pengguna.

Contoh berikut mengasumsikan bahwa sertifikat X.509 penandatanganan Anda disimpan dalam file`certificate.pem`.

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

Anda dapat menggunakan proses yang sama ini untuk memberikan beban kerja tambahan akses terprogram ke Akun AWS sumber daya Anda, jika beban kerja tidak dapat mengambil peran IAM. Prosedur ini menggunakan kebijakan ter PowerUserAccess kelola untuk menetapkan izin. Untuk mengikuti praktik terbaik hak istimewa terkecil, pertimbangkan untuk menggunakan kebijakan yang lebih ketat atau membuat kebijakan khusus yang membatasi akses hanya ke sumber daya yang diperlukan oleh program. Untuk mempelajari tentang menggunakan kebijakan yang membatasi izin pengguna ke AWS sumber daya tertentu, lihat Manajemen akses untuk AWS sumber daya danContoh kebijakan berbasis identitas IAM. Untuk menambahkan pengguna tambahan ke grup pengguna setelah dibuat, lihat Mengedit pengguna dalam grup IAM.

Gunakan otentikasi multi-faktor dengan identitas Anda

Menggunakan otentikasi multi-faktor (MFA) dengan identitas Anda adalah praktik terbaik IAM lainnya. MFA adalah lapisan keamanan tambahan yang mengharuskan pengguna untuk memberikan faktor otentikasi tambahan setelah memberikan nama pengguna dan kata sandi mereka untuk memverifikasi identitas mereka. Ini secara signifikan meningkatkan keamanan dengan membuatnya lebih sulit bagi penyerang untuk mendapatkan akses yang tidak sah, bahkan jika kata sandi pengguna dikompromikan. MFA secara luas diadopsi sebagai praktik terbaik untuk mengamankan akses ke akun online, layanan cloud, dan sumber daya sensitif lainnya. AWS mendukung MFA untuk pengguna root, pengguna IAM, pengguna di IAM Identity Center, Builder ID, dan pengguna federasi. Untuk keamanan tambahan, Anda dapat membuat kebijakan yang mengharuskan MFA dikonfigurasi sebelum mengizinkan pengguna mengakses sumber daya atau mengambil tindakan tertentu dan melampirkan kebijakan ini ke peran IAM Anda. IAM Identity Center hadir dengan MFA yang dihidupkan secara default sehingga semua pengguna di IAM Identity Center harus masuk dengan MFA selain nama pengguna dan kata sandi mereka.

> **Note**
>
> Semua Akun AWS jenis (akun mandiri, manajemen, dan anggota) memerlukan MFA untuk dikonfigurasi untuk pengguna root mereka. Pengguna harus mendaftarkan MFA dalam waktu 35 hari sejak upaya masuk pertama mereka untuk mengakses jika Konsol Manajemen AWS MFA belum diaktifkan.

Untuk informasi selengkapnya, lihat Mengkonfigurasi MFA di Pusat Identitas IAM dan. AWS Multi-factor otentikasi di IAM

Bersiaplah untuk izin hak istimewa terkecil

Menggunakan izin hak istimewa terkecil adalah rekomendasi praktik terbaik IAM. Konsep izin hak istimewa terkecil adalah untuk memberikan pengguna izin yang diperlukan untuk melakukan tugas dan tidak ada izin tambahan. Saat Anda menyiapkan, pertimbangkan bagaimana Anda akan mendukung izin hak istimewa terendah. Pengguna root, pengguna administratif, dan pengguna IAM akses darurat memiliki izin kuat yang tidak diperlukan untuk tugas sehari-hari. Saat Anda mempelajari AWS dan menguji berbagai layanan, kami sarankan Anda membuat setidaknya satu pengguna tambahan di Pusat Identitas IAM dengan izin yang lebih rendah yang dapat Anda gunakan dalam skenario yang berbeda. Anda dapat menggunakan kebijakan IAM untuk menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu dan kemudian menyambungkan ke sumber daya tersebut dengan akun yang kurang istimewa.

Jika Anda menggunakan Pusat Identitas IAM, pertimbangkan untuk menggunakan kumpulan izin Pusat Identitas IAM untuk memulai. Untuk mempelajari selengkapnya, lihat Membuat kumpulan izin di Panduan Pengguna Pusat Identitas IAM.

Jika Anda tidak menggunakan Pusat Identitas IAM, gunakan peran IAM untuk menentukan izin untuk entitas IAM yang berbeda. Untuk mempelajari selengkapnya, lihat Pembuatan peran IAM.

Peran IAM dan kumpulan izin Pusat Identitas IAM dapat menggunakan kebijakan ter AWS kelola berdasarkan fungsi pekerjaan. Untuk detail tentang izin yang diberikan oleh kebijakan ini, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

> **Important**
>
> Perlu diingat bahwa kebijakan ter AWS kelola mungkin tidak memberikan izin hak istimewa terkecil untuk kasus penggunaan tertentu karena kebijakan tersebut tersedia untuk

digunakan oleh semua pelanggan. AWS Setelah disiapkan, sebaiknya gunakan IAM Access Analyzer untuk membuat kebijakan hak istimewa terkecil berdasarkan aktivitas akses yang masuk. AWS CloudTrail Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan Penganalisis Akses IAM.

Saat Anda memulai, sebaiknya gunakan kebijakan terkel AWS ola untuk memberikan izin. Setelah periode sampel aktivitas yang telah ditentukan sebelumnya (seperti 90 hari) telah berlalu, Anda dapat meninjau layanan yang telah diakses orang dan beban kerja. Kemudian Anda dapat membuat kebijakan baru yang dikelola pelanggan dengan izin yang dikurangi untuk mengganti kebijakan AWS terkelola. Kebijakan baru harus mencakup hanya layanan yang diakses selama periode sampel. Perbarui izin Anda untuk menghapus kebijakan ter AWS kelola dan melampirkan kebijakan baru yang dikelola pelanggan yang Anda buat.

Untuk petunjuk tentang menyempurnakan izin, lihatMeninjau informasi terakhir yang diakses untuk akun Anda AWS. Setelah Anda mengidentifikasi layanan yang tidak digunakan, lihatMenghasilkan kebijakan berdasarkan aktivitas akses.

Meninjau informasi terakhir yang diakses untuk akun Anda AWS

Anda dapat melihat informasi layanan yang terakhir diakses untuk IAM menggunakan konsol IAM, AWS CLI, atau AWS API. Untuk informasi penting tentang data, izin yan diperlukan, pemecahan masalah, dan region yang didukung lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Anda dapat melihat informasi untuk jenis sumber daya berikut di IAM. Dalam setiap kejadian, informasi tersebut mencakup layanan yang diizinkan untuk periode pelaporan tertentu:

- Pengguna IAM — Lihat terakhir kali pengguna mencoba mengakses setiap layanan yang diizinkan.
- Grup IAM — Melihat informasi tentang terakhir kali anggota grup IAM mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total anggota yang mencoba mengaksesnya.
- Peran IAM — Lihat terakhir kali seseorang menggunakan peran tersebut dalam upaya mengakses setiap layanan yang diizinkan.
- Kebijakan – Melihat informasi terakhir kali pengguna atau peran mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total entitas yang mencoba mengaksesnya.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk sumber daya di IAM, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk informasi Anda. Untuk lebih detailnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Untuk informasi selengkapnya tentang informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Untuk meninjau informasi yang terakhir diakses untuk Akun AWS

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Grup pengguna, Pengguna, Peran, atau Kebijakan.
4. Pilih nama pengguna, grup pengguna, peran, atau kebijakan apa pun untuk membuka halaman Ringkasannya dan pilih tab Terakhir Diakses. Lihat informasi berikut, berdasarkan sumber daya yang Anda pilih:

   - Grup pengguna - Lihat daftar layanan yang dapat diakses oleh anggota grup pengguna. Anda juga dapat melihat kapan anggota terakhir mengakses layanan, kebijakan grup pengguna apa yang mereka gunakan, dan anggota grup pengguna mana yang membuat permintaan. Pilih nama kebijakan untuk dipelajari baik merupakan kebijakan terkelola atau kebijakan grup pengguna inline. Pilih nama anggota grup pengguna untuk melihat semua anggota grup pengguna dan kapan mereka terakhir kali mengakses layanan.
   - Pengguna – Lihat daftar layanan yang dapat diakses pengguna. Anda juga dapat melihat kapan mereka terakhir mengakses layanan, dan kebijakan apa yang saat ini terkait dengan pengguna. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola, kebijakan pengguna inline, atau kebijakan inline untuk grup pengguna.
   - Peran – Lihat daftar layanan yang dapat diakses oleh peran, kapan peran terakhir mengakses layanan, dan kebijakan apa yang digunakan. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola atau kebijakan peran selaras.

- Kebijakan – Lihat daftar layanan dengan tindakan yang diperbolehkan dalam kebijakan. Anda juga dapat melihat kapan kebijakan terakhir digunakan untuk mengakses layanan, dan entitas (pengguna atau peran) mana yang menggunakan kebijakan tersebut. Tanggal Terakhir diakses juga mencakup kapan akses diberikan ke kebijakan ini melalui kebijakan lain. Pilih nama entitas untuk dipelajari yang sudah terlampirkan dengan kebijakan ini dan kapan terakhir kali mereka mengakses layanan.

5. Di kolom Layanan tabel, pilih nama salah satu layanan yang menyertakan informasi tindakan yang terakhir diakses untuk melihat daftar tindakan manajemen yang telah dicoba diakses oleh entitas IAM. Anda dapat melihat Wilayah AWS dan stempel waktu yang menunjukkan kapan seseorang terakhir mencoba melakukan tindakan.
6. Kolom Terakhir diakses ditampilkan untuk layanan dan tindakan manajemen layanan yang menyertakan informasi tindakan yang terakhir diakses. Tinjau hasil-hasil berikut yang kemungkinan muncul dalam kolom ini. Hasil ini bervariasi tergantung pada apakah layanan atau tindakan diizinkan, diakses, dan apakah terlacak oleh AWS untuk informasi yang terakhir diakses.

   <jumlah> hari yang lalu

   Jumlah hari sejak layanan atau tindakan digunakan dalam periode pelacakan. Periode pelacakan layanan adalah selama 400 hari terakhir. Periode pelacakan untuk tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk tindakan Amazon EC2, IAM, dan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan lainnya dimulai pada 23 Mei 2023. Untuk mempelajari selengkapnya tentang melacak tanggal mulai masing-masing Wilayah AWS, lihatTempat AWS melacak informasi terakhir yang diakses.

   Tidak diakses dalam periode pelacakan

   Layanan atau tindakan yang dilacak belum digunakan oleh entitas dalam periode pelacakan.

   Anda dapat memiliki izin untuk tindakan yang tidak muncul dalam daftar. Ini dapat terjadi jika informasi pelacakan untuk tindakan saat ini tidak disertakan oleh AWS. Anda tidak diperkenankan mengambil keputusan izin hanya berdasarkan ketiadaan informasi pelacakan. Alih-alih, kami menyarankan agar Anda menggunakan informasi ini untuk menginformasikan dan mendukung strategi Anda secara keseluruhan untuk memberikan privilese paling sedikit. Periksa kebijakan Anda untuk mengonfirmasi bahwa tingkat akses sesuai.

AWS CLI

Anda dapat menggunakan AWS CLI untuk mengambil informasi tentang terakhir kali sumber daya IAM di Anda Akun AWS digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, Amazon EC2, IAM, dan Lambda. Sumber daya IAM dapat berupa pengguna, grup pengguna, peran, atau kebijakan.

- Hasilkan laporan untuk sumber daya IAM di file Akun AWS. Permintaan harus menyertakan ARN sumber daya IAM (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat detail akses baik untuk layanan maupun layanan beserta tindakan. Permintaan tersebut menghasilkan `job-id` yang kemudian dapat Anda gunakan di operasi `get-service-last-accessed-details` dan `get-service-last-accessed-details-with-entities` untuk memonitor `job-status` hingga pekerjaan selesai.

  - detail aws iam generate-service-last-accessed

  a. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

     - detail aws iam get-service-last-accessed

     Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `generate-service-last-accessed-details` operasi:

     - Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir pengguna dan ARN pengguna.
     - Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ia juga menyatakan ARN pengguna tersebut dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntities untuk memperoleh daftar semua anggota.
     - Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir peran dan ARN peran.

- Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ia juga menyatakan ARN dari entitas tersebut dan jumlah total entitas yang mencoba mengakses.

b. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini menghasilkan daftar entitas setiap ARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir kali mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - aws iam get-service-last-accessed - details-with-entities

c. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas izin IAM, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Evaluasi kebijakan untuk permintaan dalam satu akun.

   - aws iam list-policies-granting-service -akses

API

Anda dapat menggunakan AWS API untuk mengambil informasi tentang terakhir kali sumber daya IAM digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, Amazon EC2, IAM, dan Lambda. Sumber daya IAM dapat berupa pengguna, grup pengguna, peran, atau kebijakan. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat baik layanan maupun layanan beserta tindakan.

1. Buat laporan. Permintaan harus menyertakan ARN sumber daya IAM (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. Ia menghasilkan `JobId` yang kemudian dapat Anda gunakan di operasi `GetServiceLastAccessedDetails` dan

`GetServiceLastAccessedDetailsWithEntities` untuk memonitor `JobStatus` hingga pekerjaan selesai.

- GenerateServiceLastAccessedDetails

2. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetServiceLastAccessedDetails

   Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `GenerateServiceLastAccessedDetails` operasi:

   - Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir pengguna dan ARN pengguna.
   - Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ia juga menyatakan ARN pengguna tersebut dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntities untuk memperoleh daftar semua anggota.
   - Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir peran dan ARN peran.
   - Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ia juga menyatakan ARN dari entitas tersebut dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini menghasilkan daftar entitas setiap ARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir kali mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - GetServiceLastAccessedDetailsWithEntities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin

yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas izin IAM, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Evaluasi kebijakan untuk permintaan dalam satu akun.

- ListPoliciesGrantingServiceAccess

Menghasilkan kebijakan berdasarkan aktivitas akses

Anda dapat menggunakan aktivitas akses yang direkam AWS CloudTrail untuk pengguna IAM atau peran IAM agar IAM Access Analyzer menghasilkan kebijakan terkelola pelanggan untuk mengizinkan akses hanya ke layanan yang dibutuhkan pengguna dan peran tertentu.

Saat IAM Access Analyzer membuat kebijakan IAM, informasi akan dikembalikan untuk membantu Anda menyesuaikan kebijakan lebih lanjut. Dua kategori informasi dapat dihasilkan ketika kebijakan yang membuat:

- Kebijakan dengan informasi tingkat tindakan — Untuk beberapa AWS layanan, seperti Amazon EC2, IAM Access Analyzer dapat mengidentifikasi tindakan yang ditemukan dalam acara CloudTrail Anda dan mencantumkan tindakan yang digunakan dalam kebijakan yang dihasilkannya. Untuk daftar layanan yang didukung, lihatLayanan pembuatan kebijakan IAM Access Analyzer. Untuk beberapa layanan, IAM Access Analyzer meminta Anda untuk menambahkan tindakan untuk layanan ke kebijakan yang dihasilkan.
- Kebijakan dengan informasi tingkat layanan — IAM Access Analyzer menggunakan informasi yang terakhir diakses untuk membuat templat kebijakan dengan semua layanan yang baru digunakan. Saat menggunakan Konsol Manajemen AWS, kami meminta Anda untuk meninjau layanan dan menambahkan tindakan untuk menyelesaikan kebijakan.

Untuk menghasilkan kebijakan berdasarkan aktivitas akses

Dalam prosedur berikut, kami akan mengurangi izin yang diberikan ke peran agar sesuai dengan penggunaan pengguna. Saat Anda memilih pengguna, pilih pengguna yang penggunaannya mencontohkan peran tersebut. Banyak pelanggan menyiapkan akun pengguna uji dengan PowerUserizin dan kemudian meminta mereka melakukan serangkaian tugas tertentu untuk periode

waktu yang singkat untuk menentukan akses apa yang diperlukan untuk melakukan tugas-tugas tersebut,

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Pengguna dan kemudian pilih nama pengguna untuk masuk ke halaman detail pengguna.
4. Pada tab Izin, di bawah Buat kebijakan berdasarkan CloudTrail peristiwa, pilih Buat kebijakan.
5. Pada halaman Buat kebijakan, konfigurasikan item berikut:
   - Untuk Pilih periode waktu, pilih 7 hari terakhir.
   - Untuk CloudTrail jejak yang akan dianalisis, pilih Wilayah dan jejak tempat aktivitas pengguna ini direkam.
   - Pilih Buat dan gunakan peran layanan baru.
6. Pilih Buat kebijakan lalu tunggu hingga peran dibuat. Jangan me-refresh atau menjauh dari halaman konsol hingga muncul pesan notifikasi pembuatan Kebijakan yang sedang berlangsung.
7. Setelah kebijakan dibuat, Anda harus meninjau dan menyesuaikannya sesuai kebutuhan dengan akun IDs dan ARNs sumber daya. Selain itu, kebijakan yang dibuat secara otomatis mungkin tidak menyertakan informasi tingkat tindakan yang diperlukan untuk menyelesaikan kebijakan. Untuk informasi selengkapnya, lihat pembuatan kebijakan IAM Access Analyzer.

   Misalnya, Anda dapat mengedit pernyataan pertama yang menyertakan `Allow` efek dan `NotAction` elemen untuk mengizinkan hanya tindakan Amazon EC2 dan Amazon S3. Untuk melakukan ini, ganti dengan pernyataan yang memiliki ID `FullAccessToSomeServices`. Kebijakan baru Anda bisa terlihat seperti contoh kebijakan berikut.

   JSON

```
{
  "Version":"2012-10-17",
```

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

8. Untuk mendukung praktik terbaik memberikan hak istimewa paling rendah, tinjau dan perbaiki kesalahan, peringatan, atau saran yang dikembalikan selama validasi kebijakan.
9. Untuk mengurangi izin kebijakan Anda terhadap tindakan dan sumber daya tertentu, lihat acara Anda di CloudTrail riwayat Acara. Di sana Anda dapat melihat informasi terperinci tentang tindakan dan sumber daya spesifik yang telah diakses oleh pengguna Anda. Untuk informasi selengkapnya, lihat Melihat CloudTrail Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.
10. Setelah meninjau dan memvalidasi kebijakan Anda, simpan dengan nama deskriptif.
11. Arahkan ke halaman Peran dan pilih peran yang akan diambil orang saat mereka melakukan tugas yang diizinkan oleh kebijakan baru Anda.
12. Pilih tab Izin, lalu pilih Tambahkan izin dan pilih Lampirkan kebijakan.
13. Pada halaman Lampirkan kebijakan izin, di daftar Kebijakan izin lainnya, pilih kebijakan yang Anda buat, lalu pilih Lampirkan kebijakan.
14. Anda dikembalikan ke halaman Rincian Peran. Ada dua kebijakan yang melekat pada peran, kebijakan AWS terkelola Anda sebelumnya, seperti PowerUserAccess, dan kebijakan baru

Anda. Pilih kotak centang untuk kebijakan AWS terkelola, lalu pilih Hapus. Ketika diminta untuk mengonfirmasi penghapusan, pilih Hapus.

Pengguna IAM, prinsip federasi SAFL dan OIDC, serta beban kerja yang mengambil peran ini sekarang telah mengurangi akses sesuai dengan kebijakan baru yang Anda buat.

AWS CLI

Anda dapat menggunakan perintah berikut untuk menghasilkan kebijakan menggunakan AWS CLI.

Untuk membuat kebijakan

- aws accessanalyzer start-policy-generation

Untuk melihat kebijakan yang dibuat

- aws accessanalyzer get-generated-policy

Untuk membatalkan permintaan pembuatan kebijakan

- aws accessanalyzer cancel-policy-generation

Untuk melihat daftar permintaan pembuatan kebijakan

- aws accessanalyzer list-policy-generations

API

Anda dapat menggunakan operasi berikut untuk membuat kebijakan menggunakan API AWS .

Untuk membuat kebijakan

- StartPolicyGeneration

Untuk melihat kebijakan yang dibuat

- GetGeneratedPolicy

Untuk membatalkan permintaan pembuatan kebijakan

- CancelPolicyGeneration

Untuk melihat daftar permintaan pembuatan kebijakan

- ListPolicyGenerations

Menggunakan pencarian untuk menemukan sumber daya IAM

Saat Anda mengerjakan temuan akses, Anda dapat menggunakan halaman pencarian konsol IAM sebagai opsi yang lebih cepat untuk menemukan sumber daya IAM. Anda dapat mencari sumber daya menggunakan nama sumber daya sebagian atau ARN.

Console

Fitur pencarian konsol IAM dapat menemukan salah satu dari berikut ini:

- Nama entitas IAM yang sesuai dengan kata kunci pencarian Anda (untuk pengguna, grup, peran, penyedia identitas, dan kebijakan)
- Tugas yang cocok dengan kata kunci pencarian Anda

Fitur pencarian konsol IAM tidak mengembalikan informasi tentang Penganalisis Akses IAM.

Setiap baris dalam hasil pencarian adalah tautan aktif. Misalnya, Anda dapat memilih nama pengguna di hasil pencarian, yang membawa Anda ke laman detail pengguna tersebut. Atau Anda dapat memilih tautan tindakan, misalnya Buat pengguna, untuk menuju ke laman Buat Pengguna.

> **Note**
>
> Pencarian access key mengharuskan Anda mengetikkan access key ID penuh di kotak pencarian. Hasil pencarian menampilkan pengguna yang terkait dengan kunci tersebut. Dari sana Anda dapat menavigasi langsung ke halaman pengguna itu, di mana Anda dapat mengelola kunci akses.

Gunakan laman Cari di konsol IAM untuk menemukan item yang terkait dengan akun tersebut.

Untuk mencari item di konsol IAM

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Cari.
4. Di kotak Cari ketikkan kata kunci pencarian Anda.
5. Pilih tautan di daftar hasil pencarian untuk menavigasi ke bagian konsol yang sesuai.

Ikon berikut mengidentifikasi jenis-jenis IT yang ditemukan dengan pencarian:

Ikon	Deskripsi
	Pengguna IAM:
	Grup IAM
	IAM role
	Kebijakan IAM
	Tugas seperti “buat pengguna” atau “lampirkan kebijakan”
	Hasil dari kata kunci **`delete`**

Contoh frase pencarian

Anda dapat menggunakan frasa berikut dalam pencarian IAM. Ganti istilah dalam huruf miring dengan nama pengguna, grup, peran, kunci akses, kebijakan, atau penyedia identitas IAM yang sebenarnya yang ingin Anda temukan.

- ***user_name*** atau ***group_name*** atau ***role_name*** atau ***policy_name*** atau ***identity_provider_name***
- ***access_key***
- **add user *user_name* to groups** atau **add users to group *group_name***
- **remove user *user_name* from groups**
- **delete *user_name*** atau **delete *group_name*** atau **delete *role_name***, atau **delete *policy_name***, atau **delete *identity_provider_name***
- **manage access keys *user_name***
- **manage signing certificates *user_name***
- **users**
- **manage MFA for *user_name***
- **manage password for *user_name***
- **create role**
- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***
- **detach entities from *policy_name***

Praktik terbaik keamanan dan kasus penggunaan di AWS Identity and Access Management

AWS Identity and Access Management (IAM) menawarkan beberapa fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap.

Praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan spesifik Anda. Oleh karena itu, perlakukan mereka sebagai pertimbangan yang bermanfaat daripada persyaratan.

Untuk mendapatkan hasil maksimal dari IAM, luangkan waktu untuk mempelajari praktik terbaik yang direkomendasikan. Salah satu cara efektif untuk melakukan ini adalah dengan melihat bagaimana IAM digunakan dalam skenario dunia nyata untuk bekerja dengan yang lain. Layanan AWS

Topik

- Praktik terbaik keamanan di IAM
- Praktik terbaik pengguna root untuk Anda Akun AWS
- Kasus penggunaan bisnis untuk IAM

Praktik terbaik keamanan di IAM

Follow us on Twitter

Untuk membantu mengamankan AWS sumber daya Anda, ikuti praktik terbaik untuk AWS Identity and Access Management (IAM) ini.

Setelah menerapkan praktik terbaik ini, lihat Keamanan di IAM dan AWS STS topik keamanan tambahan termasuk validasi kepatuhan, ketahanan, dan keamanan infrastruktur.

Topik

- Memerlukan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara
- Memerlukan beban kerja untuk menggunakan kredenSIAL sementara dengan peran IAM untuk mengakses AWS
- Memerlukan otentikasi multi-faktor (MFA)

- Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredenSIAL jangka panjang
- Ikuti praktik terbaik untuk melindungi kredenSIAL pengguna root Anda
- Terapkan izin hak istimewa terkecil
- Memulai dengan AWS kebijakan terkelola dan bergerak menuju izin hak istimewa terendah
- Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa terkecil berdasarkan aktivitas akses
- Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredenSIAL yang tidak digunakan secara teratur
- Gunakan ketentuan dalam kebijakan IAM untuk membatasi akses lebih lanjut
- Verifikasi akses publik dan lintas akun ke sumber daya dengan IAM Access Analyzer
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk izin yang aman dan fungsional
- Buat pagar pembatas izin di beberapa akun
- Menggunakan batas izin untuk mendelegasikan manajemen izin dalam akun

Memerlukan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara

Pengguna manusia, juga dikenal sebagai identitas manusia, adalah orang, administrator, pengembang, operator, dan konsumen aplikasi Anda. Mereka harus memiliki identitas untuk mengakses AWS lingkungan dan aplikasi Anda. Pengguna manusia yang merupakan anggota organisasi Anda juga dikenal sebagai identitas tenaga kerja. Pengguna manusia juga dapat menjadi pengguna eksternal dengan siapa Anda berkolaborasi, dan yang berinteraksi dengan AWS sumber daya Anda. Mereka dapat melakukan ini melalui browser web, aplikasi klien, aplikasi seluler, atau alat baris perintah interaktif.

Memerlukan pengguna manusia Anda untuk menggunakan kredenSIAL sementara saat mengakses AWS. Anda dapat menggunakan penyedia identitas bagi pengguna manusia Anda untuk menyediakan akses federasi Akun AWS dengan mengambil peran, yang menyediakan kredentif sementara. Untuk manajemen akses terpusat, sebaiknya gunakan AWS IAM Identity Center (Pusat Identitas IAM) untuk mengelola akses ke akun dan izin di dalam akun tersebut. Anda dapat mengelola identitas pengguna dengan Pusat Identitas IAM, atau mengelola izin akses untuk identitas

pengguna di Pusat Identitas IAM dari penyedia identitas eksternal. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Anda dapat menggunakan manajer akses akun — fitur IAM yang memungkinkan Anda menetapkan peran IAM yang ada di seluruh akun organisasi Anda kepada pengguna dan grup Pusat Identitas IAM. Manajer akses akun memberi Anda akses ke set fitur peran IAM lengkap, termasuk kebijakan kepercayaan khusus, tag peran untuk ABAC, dan jalur peran yang dapat dikonfigurasi. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.

Untuk informasi lebih lanjut tentang peran, lihat Istilah dan konsep peran.

Memerlukan beban kerja untuk menggunakan kredenSIAL sementara dengan peran IAM untuk mengakses AWS

B eban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Beban kerja Anda dapat memiliki aplikasi, alat operasional, dan komponen yang memerlukan kredenSIAL untuk membuat permintaan Layanan AWS, seperti permintaan untuk membaca data dari Amazon S3.

Saat Anda membangun layanan AWS komputasi, seperti Amazon EC2 atau Lambda, AWS memberikan kredenSIAL sementara peran IAM ke sumber daya komputasi tersebut. Aplikasi yang ditulis menggunakan AWS SDK akan menemukan dan menggunakan kredenSIAL sementara ini untuk mengakses AWS sumber daya, dan tidak perlu mendistribusikan kredenSIAL berumur panjang untuk pengguna IAM ke beban kerja Anda yang sedang berjalan. AWS

Beban kerja yang berjalan di luar AWS, seperti server lokal Anda, server dari penyedia cloud lain, atau integrasi berkelanjutan terkelola dan platform pengiriman berkelanjutan (CI/CD), masih dapat menggunakan kredenSIAL sementara. Namun, Anda harus mengirimkan kredenSIAL sementara ini ke beban kerja Anda. Berikut ini adalah cara Anda dapat mengirimkan kredenSIAL sementara ke beban kerja Anda:

- Anda dapat menggunakan IAM Roles Anywhere untuk meminta AWS kredenSIAL sementara untuk beban kerja Anda menggunakan X.509 Sertifikat dari infrastruktur kunci publik (PKI) Anda.
- Anda dapat memanggil AWS STS `AssumeRoleWithSAML` API untuk meminta AWS kredenSIAL sementara untuk beban kerja Anda menggunakan pernyataan SAML dari penyedia identitas eksternal (IdP) yang dikonfigurasi dalam Anda. Akun AWS
- Anda dapat memanggil AWS STS `AssumeRoleWithWebIdentity` API untuk meminta AWS kredenSIAL sementara untuk beban kerja Anda menggunakan token web JSON (JWT) dari IdP yang dikonfigurasi dalam Anda. Akun AWS

- Anda dapat meminta AWS kredenSIAL sementara dari perangkat IoT Anda menggunakan otentikasi Mutual Transport Layer Security (MTLS) menggunakan AWS IoT Core.

Beberapa Layanan AWS juga mendukung integrasi untuk memberikan kredenSIAL sementara ke beban kerja Anda yang berjalan di luar: AWS

- Amazon Elastic Container Service (Amazon ECS) Anywhere memungkinkan Anda menjalankan tugas Amazon ECS pada sumber daya komputasi Anda sendiri, dan memberikan AWS kredentif sementara ke tugas Amazon ECS yang berjalan pada sumber daya komputasi tersebut.
- Amazon Elastic Kubernetes Service Hybrid Nodes memungkinkan Anda menggabungkan sumber daya komputasi yang berjalan di luar AWS sebagai node ke cluster Amazon EKS. Amazon EKS dapat mengirimkan kredenSIAL sementara ke pod Amazon EKS yang berjalan di sumber daya komputasi Anda.
- AWS Systems ManagerHybrid Activationsmemungkinkan Anda mengelola sumber daya komputasi yang berjalan di luar AWS penggunaan SSM, dan memberikan AWS kredentif sementara ke agen SSM yang berjalan di sumber daya komputasi Anda.

Memerlukan otentikasi multi-faktor (MFA)

Sebaiknya gunakan peran IAM untuk pengguna manusia dan beban kerja yang mengakses AWS sumber daya Anda sehingga mereka menggunakan kredenSIAL sementara. Namun, untuk skenario di mana Anda memerlukan pengguna IAM atau pengguna root di akun Anda, memerlukan MFA untuk keamanan tambahan. Dengan MFA, pengguna memiliki perangkat yang menghasilkan respons terhadap tantangan autentikasi. KredenSIAL setiap pengguna dan respons yang dihasilkan perangkat diperlukan untuk menyelesaikan proses masuk. Untuk informasi selengkapnya, lihat AWS Multi-factor otentikasi di IAM. Sebaiknya gunakan MFA yang tahan phishing seperti kunci sandi dan kunci keamanan jika memungkinkan. Untuk informasi selengkapnya, lihat Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS.

Jika Anda menggunakan Pusat Identitas IAM untuk manajemen akses terpusat bagi pengguna manusia, Anda dapat menggunakan kemampuan MFA Pusat Identitas IAM saat sumber identitas Anda dikonfigurasi dengan penyimpanan identitas Pusat Identitas IAM, Microsoft AD Terkel AWS ola, atau Konektor AD. Untuk informasi selengkapnya tentang MFA di Pusat Identitas IAM, lihat Multi-factor otentikasi di Panduan AWS IAM Identity Center Pengguna.

Perbarui kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredenSIAL jangka panjang

Jika memungkinkan, kami sarankan mengandalkan kredenSIAL sementara daripada membuat kredenSIAL jangka panjang seperti kunci akses. Namun, untuk skenario di mana Anda memerlukan pengguna IAM dengan akses terprogram dan kredenSIAL jangka panjang, kami sarankan Anda memperbarui kunci akses saat diperlukan, seperti ketika karyawan meninggalkan perusahaan Anda. Kami menyarankan Anda menggunakan informasi terakhir yang digunakan akses IAM untuk memperbarui dan menghapus kunci akses dengan aman. Untuk informasi selengkapnya, lihat Perbarui kunci akses.

Ada kasus penggunaan khusus yang memerlukan kredenSIAL jangka panjang dengan pengguna IAM di AWS dalamnya. Beberapa kasus penggunaan meliputi yang berikut:

- Beban kerja yang tidak dapat menggunakan peran IAM — Anda mungkin menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan peran IAM untuk memberikan kredenSIAL sementara, seperti untuk WordPress plugin. Dalam situasi ini, gunakan kunci akses jangka panjang pengguna IAM untuk beban kerja tersebut untuk diautentikasi. AWS
- Third-party AWS klien — Jika Anda menggunakan alat yang tidak mendukung akses dengan Pusat Identitas IAM, seperti AWS klien pihak ketiga atau vendor yang tidak di-host AWS, gunakan kunci akses jangka panjang pengguna IAM.
- AWS CodeCommit akses — Jika Anda menggunakan CodeCommit untuk menyimpan kode Anda, Anda dapat menggunakan pengguna IAM dengan kunci SSH atau kredenSIAL khusus layanan untuk mengotentikasi CodeCommit ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di Pusat Identitas IAM untuk otentikasi normal. Pengguna di Pusat Identitas IAM adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengkonfigurasi pengguna IAM, Anda dapat mengonfigurasi utilitasgit-remote-codecommit. Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihatKredensi IAM untuk: Kredensi CodeCommit Git, kunci SSH, dan kunci akses AWS. Untuk informasi selengkapnya tentang mengonfigurasi git-remote-codecommit utilitas, lihat Menghubungkan ke AWS CodeCommit repositori dengan kredenSIAL berputar di Panduan AWS CodeCommit Pengguna.
- Akses Amazon Keyspaces (untuk Apache Cassandra) — Dalam situasi di mana Anda tidak dapat menggunakan pengguna di Pusat Identitas IAM, seperti untuk tujuan pengujian kompatibilitas

Cassandra, Anda dapat menggunakan pengguna IAM dengan kredenSIAL khusus layanan untuk mengotentikasi dengan Amazon Keyspaces. Pengguna di Pusat Identitas IAM adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi Anda Akun AWS atau ke aplikasi cloud Anda. Anda juga dapat terhubung ke Amazon Keyspaces menggunakan kredenSIAL sementara. Untuk informasi selengkapnya, lihat Menggunakan kredenSIAL sementara untuk menyambung ke ruang kunci Amazon menggunakan peran IAM dan plugin SIGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

Ikuti praktik terbaik untuk melindungi kredenSIAL pengguna root Anda

Saat Anda membuat Akun AWS, Anda membuat kredenSIAL pengguna root untuk masuk ke Konsol Manajemen AWS. Lindungi kredentif pengguna root Anda dengan cara yang sama seperti Anda melindungi informasi pribadi sensitif lainnya. Untuk lebih memahami cara mengamankan dan menskalakan proses pengguna root Anda, lihatPraktik terbaik pengguna root untuk Anda Akun AWS.

Terapkan izin hak istimewa terkecil

Saat Anda menetapkan izin dengan kebijakan IAM, berikan hanya izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Anda dapat memulai dengan izin luas saat Anda menjelajahi izin yang diperlukan untuk beban kerja atau kasus penggunaan Anda. Saat kasus penggunaan Anda matang, Anda dapat bekerja untuk mengurangi izin yang Anda berikan untuk bekerja menuju hak istimewa yang paling sedikit. Untuk informasi selengkapnya tentang menggunakan IAM untuk menerapkan izin, lihatKebijakan dan izin di AWS Identity and Access Management.

Memulai dengan AWS kebijakan terkelola dan bergerak menuju izin hak istimewa terendah

Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan terkel AWS ola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Perlu diingat bahwa kebijakan ter AWS kelola mungkin tidak memberikan izin hak istimewa terkecil untuk kasus penggunaan tertentu karena kebijakan tersebut tersedia untuk digunakan oleh semua pelanggan. AWS Oleh karena itu, kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat AWS kebijakan terkelola. Untuk informasi

selengkapnya AWS tentang kebijakan terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Gunakan IAM Access Analyzer untuk menghasilkan kebijakan hak istimewa terkecil berdasarkan aktivitas akses

Untuk hanya memberikan izin yang diperlukan untuk melakukan tugas, Anda dapat membuat kebijakan berdasarkan aktivitas akses yang masuk AWS CloudTrail. IAM Access Analyzer menganalisis layanan dan tindakan yang digunakan peran IAM Anda, dan kemudian menghasilkan kebijakan terperinci yang dapat Anda gunakan. Setelah menguji setiap kebijakan yang dihasilkan, Anda dapat menerapkan kebijakan tersebut ke lingkungan produksi Anda. Ini memastikan bahwa Anda hanya memberikan izin yang diperlukan untuk beban kerja Anda. Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan Penganalisis Akses IAM.

Tinjau dan hapus pengguna, peran, izin, kebijakan, dan kredenSIAL yang tidak digunakan secara teratur

Anda mungkin memiliki pengguna, peran, izin, kebijakan, atau kredensi IAM yang tidak lagi Anda perlukan di. Akun AWS IAM menyediakan informasi terakhir yang diakses untuk membantu Anda mengidentifikasi pengguna, peran, izin, kebijakan, dan kredentif yang tidak lagi Anda perlukan sehingga Anda dapat menghapusnya. Ini membantu Anda mengurangi jumlah pengguna, peran, izin, kebijakan, dan kredenSIAL yang harus Anda pantau. Anda juga dapat menggunakan informasi ini untuk menyempurnakan kebijakan IAM agar lebih mematuhi izin hak istimewa terendah. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Gunakan ketentuan dalam kebijakan IAM untuk membatasi akses lebih lanjut

Anda dapat menentukan kondisi di mana pernyataan kebijakan berlaku. Dengan begitu, Anda dapat memberikan akses ke tindakan dan sumber daya, tetapi hanya jika permintaan akses memenuhi persyaratan tertentu. Misalnya, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan TLS. Anda juga dapat menggunakan kondisi untuk memberikan akses ke tindakan layanan, tetapi hanya jika digunakan melalui tindakan tertentu Layanan AWS, seperti CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM JSON: Kondisi.

Verifikasi akses publik dan lintas akun ke sumber daya dengan IAM Access Analyzer

Sebelum Anda memberikan izin untuk akses publik atau lintas akun AWS, sebaiknya verifikasi apakah akses tersebut diperlukan. Anda dapat menggunakan IAM Access Analyzer untuk membantu Anda melihat pratinjau dan menganalisis akses publik dan lintas akun untuk jenis sumber daya yang didukung. Anda melakukan ini dengan meninjau temuan yang dihasilkan oleh IAM Access Analyzer. Temuan ini membantu Anda memverifikasi bahwa kontrol akses sumber daya memberikan akses yang Anda harapkan. Selain itu, saat memperbarui izin publik dan lintas akun, Anda dapat memverifikasi efek perubahan sebelum menerapkan kontrol akses baru ke sumber daya Anda. IAM Access Analyzer juga memantau jenis sumber daya yang didukung secara terus menerus dan menghasilkan temuan untuk sumber daya yang memungkinkan akses publik atau lintas akun. Untuk informasi selengkapnya, lihat Pr atinjau akses dengan API Penganalisis Akses IAM.

Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk izin yang aman dan fungsional

Validasi kebijakan yang Anda buat untuk memastikan bahwa kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. Anda dapat memvalidasi kebijakan Anda dengan menggunakan validasi kebijakan IAM Access Analyzer. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Saat Anda membuat kebijakan baru atau mengedit kebijakan yang ada di konsol, IAM Access Analyzer memberikan rekomendasi untuk membantu Anda menyempurnakan dan memvalidasi kebijakan sebelum menyimpannya. Selain itu, kami menyarankan Anda meninjau dan memvalidasi semua kebijakan yang ada. Untuk informasi selengkapnya, lihat validasi kebijakan IAM Access Analyzer. Untuk informasi selengkapnya tentang pemeriksaan kebijakan yang disediakan oleh IAM Access Analyzer, lihat referensi pemeriksaan kebijakan Penganalisis Akses IAM.

Buat pagar pembatas izin di beberapa akun

Saat Anda menskalakan beban kerja, pisahkan dengan menggunakan beberapa akun yang dikelola dengan AWS Organizations. Sebaiknya gunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk membuat pagar pembatas izin untuk mengontrol akses untuk semua kepala sekolah (peran dan pengguna IAM) di seluruh akun Anda. Sebaiknya gunakan kebijakan kontrol AWS Organizations sumber daya (RCP) untuk membuat pagar pembatas izin untuk mengontrol akses AWS sumber daya di seluruh organisasi Anda. SCP dan RCP adalah jenis kebijakan organisasi

yang dapat Anda gunakan untuk mengelola izin di organisasi Anda di tingkat AWS organisasi, unit organisasi (OU), atau akun.

Namun, SCP dan RCP saja tidak cukup untuk memberikan izin kepada kepala sekolah dan sumber daya di organisasi Anda. Tidak ada izin yang diberikan oleh SCP dan RCP. Untuk memberikan izin, Anda harus melampirkan kebijakan berbasis identitas atau berbasis sumber daya ke pengguna IAM, peran IAM, atau sumber daya di akun Anda. Untuk informasi selengkapnya, lihat Blok bangunan SRA — AWS Organizations, akun, dan pagar pembatas.

Menggunakan batas izin untuk mendelegasikan manajemen izin dalam akun

Dalam beberapa skenario, Anda mungkin ingin mendelegasikan pengelolaan izin dalam akun kepada orang lain. Misalnya, Anda dapat mengizinkan pengembang untuk membuat dan mengelola peran untuk beban kerja mereka. Saat Anda mendelegasikan izin kepada orang lain, gunakan batas izin untuk menetapkan izin maksimum yang Anda delegasikan. Batas izin adalah fitur lanjutan untuk menggunakan kebijakan terkelola untuk menetapkan izin maksimum yang dapat diberikan kebijakan berbasis identitas ke peran IAM. Batas izin tidak memberikan izin dengan sendirinya. Lihat informasi yang lebih lengkap di Batas izin untuk entitas IAM.

Praktik terbaik pengguna root untuk Anda Akun AWS

Saat pertama kali membuat Akun AWS, Anda mulai dengan kumpulan kredenSIAL default dengan akses lengkap ke semua AWS sumber daya di akun Anda. Identitas ini disebut pengguna Akun AWS root. Kami sangat menyarankan Anda untuk tidak mengakses pengguna Akun AWS root kecuali Anda memiliki tugas yang memerlukan kredenSIAL pengguna root. Anda perlu mengamankan kredentif pengguna root dan mekanisme pemulihan akun Anda untuk membantu memastikan Anda tidak mengekspos kredentif yang sangat istimewa untuk penggunaan yang tidak sah.

Untuk beberapa yang Akun AWS dikelola AWS Organizations, sebaiknya hapus kredentif pengguna root dari akun anggota untuk membantu mencegah penggunaan yang tidak sah. Anda dapat menghapus kata sandi pengguna root, kunci akses, sertifikat penandatanganan, dan menonaktifkan dan menghapus otentikasi multi-faktor (MFA). Akun anggota tidak dapat masuk ke pengguna root mereka atau melakukan pemulihan kata sandi untuk pengguna root mereka. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

Alih-alih mengakses pengguna root, buat pengguna administratif untuk tugas sehari-hari.

- Jika Anda memiliki yang baru Akun AWS, lihatMenyiapkan Akun AWS.

- Untuk beberapa melalui yang Akun AWS dikelola AWS Organizations, lihat Menyiapkan Akun AWS akses untuk pengguna administratif Pusat Identitas IAM.

Dengan pengguna administratif Anda, Anda kemudian dapat membuat identitas tambahan untuk pengguna yang membutuhkan akses ke sumber daya di Anda Akun AWS. Kami sangat menyarankan Anda meminta pengguna untuk mengotentikasi dengan kredenSIAL sementara saat mengakses AWS.

- Untuk satu, mandiri Akun AWS, gunakan Peran IAM untuk membuat identitas di akun Anda dengan izin tertentu. Peran dimaksudkan untuk diambil oleh siapa saja yang membutuhkannya. Selain itu, peran tidak memiliki kredenSIAL jangka panjang standar, seperti kata sandi atau kunci akses, yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran. Tidak seperti peran IAM, Pengguna IAM: memiliki kredenSIAL jangka panjang seperti kata sandi dan kunci akses. Jika memungkinkan, praktik terbaik merekomendasikan mengandalkan kredenSIAL sementara daripada membuat pengguna IAM yang memiliki kredenSIAL jangka panjang seperti kata sandi dan kunci akses.
- Untuk beberapa melalui Akun AWS yang dikelola AWS Organizations, gunakan pengguna tenaga kerja Pusat Identitas IAM. Dengan Pusat Identitas IAM, Anda dapat mengelola pengguna secara terpusat di seluruh akun Anda Akun AWS dan izin di dalam akun tersebut. Kelola identitas pengguna Anda dengan Pusat Identitas IAM atau dari penyedia identitas eksternal. Untuk informasi selengkapnya, lihat Apa itu AWS IAM Identity Center dalam Panduan Pengguna AWS IAM Identity Center .

Topik

- Amankan kredentif pengguna root Anda untuk mencegah penggunaan yang tidak sah
- Gunakan kata sandi pengguna root yang kuat untuk membantu melindungi akses
- Amankan login pengguna root Anda dengan otentikasi multi-faktor (MFA)
- Jangan membuat kunci akses untuk pengguna root
- Gunakan persetujuan multi-orang untuk masuk pengguna root jika memungkinkan
- Gunakan alamat email grup untuk kredensional pengguna root
- Batasi akses ke mekanisme pemulihan akun
- Amankan Anda AWS Organizations kredenSIAL pengguna root akun
- Memantau akses dan penggunaan

Amankan kredentif pengguna root Anda untuk mencegah penggunaan yang tidak sah

Amankan kredentif pengguna root Anda dan gunakan hanya untuk tugas yang memerlukannya. Untuk membantu mencegah penggunaan yang tidak sah, jangan bagikan kata sandi pengguna root, MFA, kunci akses, pasangan CloudFront kunci, atau sertifikat penandatanganan dengan siapa pun, kecuali mereka yang memiliki kebutuhan bisnis yang ketat untuk mengakses pengguna root.

Jangan menyimpan kata sandi pengguna root dengan alat yang bergantung Layanan AWS pada akun yang diakses menggunakan kata sandi yang sama. Jika Anda kehilangan atau lupa kata sandi pengguna root Anda, Anda tidak akan dapat mengakses alat ini. Sebaiknya Anda memprioritaskan ketahanan dan mempertimbangkan untuk mewajibkan dua orang atau lebih untuk mengotorisasi akses ke lokasi penyimpanan. Akses ke kata sandi atau lokasi penyimpanannya harus dicatat dan dipantau.

Gunakan kata sandi pengguna root yang kuat untuk membantu melindungi akses

Kami menyarankan Anda menggunakan kata sandi yang kuat dan unik. Alat seperti pengelola kata sandi dengan algoritma pembuatan kata sandi yang kuat dapat membantu Anda mencapai tujuan ini. AWS mengharuskan kata sandi Anda memenuhi persyaratan berikut:

- Itu harus memiliki minimal 8 karakter dan maksimum 128 karakter.
- Ini harus mencakup minimal tiga dari campuran jenis karakter berikut: huruf besar, huruf kecil, angka, dan! @ # $% ^ & * () <> [] {} | _+-= simbol.
- Itu tidak boleh identik dengan Akun AWS nama atau alamat email Anda.

Untuk informasi selengkapnya, lihat Ubah kata sandi untuk Pengguna root akun AWS.

Amankan login pengguna root Anda dengan otentikasi multi-faktor (MFA)

Karena pengguna root dapat melakukan tindakan istimewa, sangat penting untuk menambahkan MFA untuk pengguna root sebagai faktor otentikasi kedua selain alamat email dan kata sandi sebagai kredenSIAL masuk. Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Akun AWS root Anda.

Kami sangat menyarankan untuk mengaktifkan beberapa perangkat MFA untuk kredenSIAL pengguna root Anda untuk memberikan fleksibilitas dan ketahanan tambahan dalam strategi

keamanan Anda. Semua Akun AWS jenis (akun mandiri, manajemen, dan anggota) memerlukan MFA untuk dikonfigurasi untuk pengguna root mereka. Pengguna harus mendaftarkan MFA dalam waktu 35 hari sejak upaya masuk pertama mereka untuk mengakses Konsol Manajemen AWS jika MFA belum diaktifkan.

- Kunci keamanan perangkat keras bersertifikat FIDO disediakan oleh penyedia pihak ketiga. Untuk informasi selengkapnya, lihat Meng aktifkan kunci keamanan FIDO untuk pengguna Akun AWS root.
- Perangkat keras yang menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali berbasis waktu (TOTP). Untuk informasi selengkapnya, lihat Meng aktifkan token TOTP perangkat keras untuk pengguna Akun AWS root.
- Aplikasi otentikator virtual yang berjalan di telepon atau perangkat lain dan meniru perangkat fisik. Untuk informasi selengkapnya, lihat Meng aktifkan perangkat MFA virtual untuk pengguna Akun AWS root Anda.

Jangan membuat kunci akses untuk pengguna root

Tombol akses memungkinkan Anda menjalankan AWS perintah di Command Line Interface (AWS CLI) atau menggunakan operasi API dari salah satu AWS SDK. Kami sangat menyarankan agar Anda tidak membuat kunci akses untuk pengguna root Anda karena pengguna root memiliki akses penuh ke semua Layanan AWS dan sumber daya di akun, termasuk informasi penagihan.

Karena hanya beberapa tugas yang memerlukan pengguna root dan Anda biasanya jarang melakukan tugas-tugas tersebut, sebaiknya masuk Konsol Manajemen AWS ke untuk melakukan tugas pengguna root. Untuk akses terprogram, alih-alih membuat kunci akses untuk pengguna root, gunakan `aws login` perintah dengan kredenSIAL root Anda untuk mengotentikasi ke SDK AWS CLI dan. Pendekatan ini menyediakan kredenSIAL sementara yang diputar secara otomatis yang meningkatkan postur keamanan Anda sambil menghilangkan kebutuhan untuk mengelola kunci akses jangka panjang. Sebelum membuat kunci akses, tinjauAlternatif untuk kunci akses jangka panjang.

Gunakan persetujuan multi-orang untuk masuk pengguna root jika memungkinkan

Pertimbangkan untuk menggunakan persetujuan multi-orang untuk memastikan bahwa tidak ada orang yang dapat mengakses MFA dan kata sandi untuk pengguna root. Beberapa perusahaan menambahkan lapisan keamanan tambahan dengan mengatur satu kelompok administrator dengan

akses ke kata sandi, dan kelompok administrator lain dengan akses ke MFA. Satu anggota dari setiap grup harus berkumpul untuk masuk sebagai pengguna root.

Gunakan alamat email grup untuk kredensional pengguna root

Gunakan alamat email yang dikelola oleh bisnis Anda dan meneruskan pesan yang diterima langsung ke sekelompok pengguna. Jika AWS harus menghubungi pemilik akun, pendekatan ini mengurangi risiko keterlambatan dalam merespons, bahkan jika individu sedang berlibur, sakit, atau telah meninggalkan bisnis. Alamat email yang digunakan untuk pengguna root tidak boleh digunakan untuk tujuan lain.

Batasi akses ke mekanisme pemulihan akun

Pastikan Anda mengembangkan proses untuk mengelola mekanisme pemulihan kredensi pengguna root jika Anda memerlukan akses ke sana selama keadaan darurat seperti pengambilalihan akun administratif Anda.

- Pastikan Anda memiliki akses ke kotak masuk email pengguna root Anda sehingga Anda dapat mengatur ulang kata sandi pengguna root yang hilang atau terlupakan.
- Jika MFA untuk pengguna Akun AWS root Anda hilang, rusak, atau tidak berfungsi, Anda dapat masuk menggunakan MFA lain yang terdaftar ke kredensional pengguna root yang sama. Jika Anda kehilangan akses ke semua MFA Anda, Anda memerlukan nomor telepon dan email yang digunakan untuk mendaftarkan akun Anda, agar terkini dan dapat diakses untuk memulihkan MFA Anda. Untuk detailnya, lihat Memulihkan perangkat MFA pengguna root.
- Jika Anda memilih untuk tidak menyimpan kata sandi pengguna root dan MFA, maka nomor telepon yang terdaftar di akun Anda dapat digunakan sebagai cara alternatif untuk memulihkan kredenSIAL pengguna root. Pastikan Anda memiliki akses ke nomor telepon kontak, perbarui nomor telepon, dan batasi siapa yang memiliki akses untuk mengelola nomor telepon.

Tidak ada orang yang boleh memiliki akses ke kotak masuk email dan nomor telepon karena keduanya adalah saluran verifikasi untuk memulihkan kata sandi pengguna root Anda. Penting untuk memiliki dua kelompok individu yang mengelola saluran ini. Satu grup memiliki akses ke alamat email utama Anda dan grup lain memiliki akses ke nomor telepon utama untuk memulihkan akses ke akun Anda sebagai pengguna root.

Amankan Anda AWS Organizations kredenSIAL pengguna root akun

Saat Anda beralih ke strategi multi-akun AWS Organizations, masing-masing Akun AWS memiliki kredenSIAL pengguna root sendiri yang perlu Anda amankan. Akun yang Anda gunakan untuk membuat organisasi adalah akun manajemen dan akun lainnya di organisasi Anda adalah akun anggota.

Amankan kredentif pengguna root untuk akun manajemen

AWS mengharuskan Anda mendaftarkan MFA untuk pengguna root dari akun manajemen organisasi Anda. Pendaftaran MFA harus diselesaikan selama upaya masuk pertama atau dalam masa tenggang 35 hari. Jika MFA tidak diaktifkan dalam waktu ini, pendaftaran akan diperlukan sebelum Anda dapat mengakses. Konsol Manajemen AWS Untuk informasi selengkapnya, lihat Multi-factor otentikasi untuk Pengguna root akun AWS.

Amankan kredentif pengguna root untuk akun anggota

Jika Anda menggunakan AWS Organizations untuk mengelola beberapa akun, ada dua strategi yang dapat Anda ambil untuk mengamankan akses pengguna root di akun Anda AWS Organizations.

- Sentralisasikan akses root dan hapus kredenSIAL pengguna root dari akun anggota. Hapus kredentif pengguna root, kunci akses, sertifikat penandatanganan, dan nonaktifkan serta hapus otentikasi multi-faktor (MFA). Ketika strategi ini digunakan, akun anggota tidak dapat masuk ke pengguna root mereka atau melakukan pemulihan kata sandi untuk pengguna root mereka. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.
- Amankan kredentif pengguna root AWS Organizations akun Anda dengan MFA untuk meningkatkan keamanan akun. Untuk informasi selengkapnya, lihat Multi-factor otentikasi untuk Pengguna root akun AWS.

Untuk detailnya, lihat Mengakses akun anggota di organisasi Anda di Panduan AWS Organizations Pengguna.

Tetapkan kontrol keamanan preventif di AWS Organizations menggunakan kebijakan kontrol layanan (SCP)

Jika akun anggota di organisasi Anda mengaktifkan kredentif pengguna root, Anda dapat menerapkan SCP untuk membatasi akses ke pengguna root akun anggota. Menolak semua tindakan pengguna root di akun anggota Anda, kecuali untuk tindakan khusus root tertentu, membantu

mencegah akses yang tidak sah. Untuk detailnya, lihat Menggunakan SCP untuk membatasi apa yang dapat dilakukan pengguna root di akun anggota Anda.

Memantau akses dan penggunaan

Sebaiknya gunakan mekanisme pelacakan saat ini untuk memantau, memperingatkan, dan melaporkan login dan penggunaan kredentif pengguna root, termasuk peringatan yang mengumumkan login dan penggunaan pengguna root. Layanan berikut dapat membantu memastikan bahwa penggunaan kredensi pengguna root dilacak dan melakukan pemeriksaan keamanan yang dapat membantu mencegah penggunaan yang tidak sah.

> **Note**
>
> CloudTrail mencatat peristiwa masuk yang berbeda untuk pengguna root dan sesi pengguna root yang memiliki hak istimewa. Sesi istimewa ini memungkinkan tugas yang memerlukan kredenSIAL pengguna root dilakukan di akun anggota di organisasi Anda. Anda dapat menggunakan acara masuk untuk mengidentifikasi tindakan yang diambil oleh akun manajemen atau administrator yang didelegasikan menggunakan `sts:AssumeRoot`. Untuk informasi selengkapnya, lihat Lacak tugas istimewa di CloudTrail.

- Jika Anda ingin diberi tahu tentang aktivitas masuk pengguna root di akun Anda, Anda dapat memanfaatkan Amazon CloudWatch untuk membuat aturan Acara yang mendeteksi kapan kredenSIAL pengguna root digunakan dan memicu pemberitahuan kepada administrator keamanan Anda. Untuk detailnya, lihat Mem antau dan memberitah Akun AWS ukan aktivitas pengguna root.
- Jika Anda ingin menyiapkan notifikasi untuk mengingatkan Anda tentang tindakan pengguna root yang disetujui, Anda dapat memanfaatkan Amazon EventBridge bersama dengan Amazon SNS untuk menulis EventBridge aturan untuk melacak penggunaan pengguna root untuk tindakan tertentu dan memberi tahu Anda menggunakan topik Amazon SNS. Sebagai contoh, lihat Men girim notifikasi saat objek Amazon S3 dibuat.
- Jika Anda sudah menggunakan layanan deteksi ancaman GuardDuty sebagai layanan, Anda dapat memperluas kemampuannya untuk memberi tahu Anda saat kredenSIAL pengguna root digunakan di akun Anda.

Peringatan harus menyertakan, tetapi tidak terbatas pada, alamat email untuk pengguna root. Miliki prosedur untuk cara menanggapi peringatan sehingga personel yang menerima peringatan akses

pengguna root memahami cara memvalidasi bahwa akses pengguna root diharapkan, dan cara meningkatkan jika mereka yakin bahwa insiden keamanan sedang berlangsung. Untuk contoh cara mengonfigurasi peringatan, lihat Meman tau dan memberitah Akun AWS ukan aktivitas pengguna root.

Mengevaluasi kepatuhan MFA pengguna root

Layanan berikut dapat membantu mengevaluasi kepatuhan MFA untuk kredenSIAL pengguna root.

> MFA-related aturan mengembalikan tidak sesuai jika Anda mengikuti praktik terbaik untuk menghapus kredentif pengguna root.
>
> Sebaiknya hapus kredentif pengguna root dari akun anggota di organisasi Anda untuk membantu mencegah penggunaan yang tidak sah. Setelah Anda menghapus kredentif pengguna root, termasuk MFA, akun anggota ini dievaluasi sebagai tidak berlaku.

- AWS Config menyediakan aturan untuk memantau kepatuhan dengan praktik terbaik pengguna root. Anda dapat menggunakan aturan AWS Config terkelola untuk membantu Anda menerapkan MFA untuk kredenSIAL pengguna root. AWS Config juga dapat mengidentifikasi kunci akses untuk pengguna root.
- Security Hub CSPM memberi Anda pandangan komprehensif tentang status keamanan Anda AWS dan membantu Anda menilai AWS lingkungan Anda terhadap standar industri keamanan dan praktik terbaik, seperti memiliki MFA pada pengguna root dan tidak memiliki kunci akses pengguna root. Untuk detail tentang aturan yang tersedia, lihat AWS Identity and Access Management kontrol di Panduan Pengguna CSPM Hub Keamanan.
- Trusted Advisor menyediakan pemeriksaan keamanan sehingga Anda tahu jika MFA tidak diaktifkan pada akun pengguna root. Untuk informasi selengkapnya, lihat MFA pada Akun Root di Panduan AWS Pengguna Dukungan.

Jika Anda perlu melaporkan masalah keamanan di akun Anda, lihat L aporkan Email yang Mencurigakan atau Pelaporan Kerentanan. Atau, Anda dapat Hu bungi AWS untuk bantuan dan bimbingan tambahan.

Kasus penggunaan bisnis untuk IAM

Kasus penggunaan bisnis sederhana untuk IAM dapat membantu Anda memahami cara-cara dasar Anda dapat menerapkan layanan untuk mengontrol AWS akses yang dimiliki pengguna Anda. Kasus penggunaan dijelaskan secara umum, tanpa mekanisme bagaimana Anda akan menggunakan IAM API untuk mencapai hasil yang Anda inginkan.

Kasus penggunaan ini melihat pada dua cara umum perusahaan fiktif bernama Example Corp dapat menggunakan IAM. Skenario pertama mempertimbangkan Amazon Elastic Compute Cloud (Amazon EC2). Kedua, Amazon Simple Storage Service (Amazon S3).

Untuk informasi selengkapnya tentang menggunakan IAM dengan layanan lain dari AWS, lihatAWS layanan yang bekerja dengan IAM.

Topik

- Pengaturan awal example corp
- Gunakan kasus untuk IAM dengan Amazon EC2
- Gunakan kasus untuk IAM dengan Amazon S3

Pengaturan awal example corp

Nikki Wolf dan Mateo Jackson adalah pendiri Example Corp. Setelah memulai perusahaan, mereka membuat Akun AWS dan mengatur AWS IAM Identity Center(Pusat Identitas IAM) untuk membuat akun administratif untuk digunakan dengan sumber daya mereka. AWS Saat Anda mengatur akses akun untuk pengguna administratif, Pusat Identitas IAM membuat peran IAM yang sesuai. Peran ini, yang dikendalikan oleh Pusat Identitas IAM, dibuat di yang relevan Akun AWS, dan kebijakan yang ditentukan dalam kumpulan AdministratorAccess izin dilampirkan ke peran tersebut.

Karena mereka sekarang memiliki akun administrator, Nikki dan Mateo tidak perlu lagi menggunakan pengguna root mereka untuk mengaksesnya. Akun AWS Mereka berencana untuk hanya menggunakan pengguna root untuk menyelesaikan tugas yang hanya dapat dilakukan oleh pengguna root. Setelah meninjau praktik terbaik keamanan, mereka mengonfigurasi otentikasi multi-faktor (MFA) untuk kredentif pengguna root mereka dan memutuskan cara melindungi kredentif pengguna root mereka.

Seiring pertumbuhan perusahaan mereka, mereka mempekerjakan karyawan untuk bekerja sebagai pengembang, admin, penguji, manajer, dan administrator sistem. Nikki bertanggung jawab atas

operasi, sementara Mateo mengelola tim teknik. Mereka menyiapkan Server Domain Direktori Aktif untuk mengelola akun karyawan dan mengelola akses ke sumber daya internal perusahaan.

Untuk memberi karyawan mereka akses ke AWS sumber daya, mereka menggunakan Pusat Identitas IAM untuk menghubungkan Direktori Aktif perusahaan mereka ke sumber daya mereka Akun AWS.

Karena mereka menghubungkan Active Directory ke IAM Identity Center, pengguna, grup, dan keanggotaan grup disinkronkan dan ditentukan. Mereka harus menetapkan kumpulan izin dan peran ke grup yang berbeda untuk memberi pengguna tingkat akses yang benar ke AWS sumber daya. Mereka menggunakan AWS kebijakan terkelola untuk fungsi pekerjaan Konsol Manajemen AWS untuk membuat kumpulan izin ini:

- Administrator
- Penagihan
- Pengembang
- Administrator jaringan
- Administrator database
- Administrator sistem
- Mendukung pengguna

Kemudian mereka menetapkan kumpulan izin ini ke peran yang ditetapkan ke grup Direktori Aktif mereka.

Untuk panduan langkah demi langkah yang menjelaskan konfigurasi awal Pusat Identitas IAM, lihat Memulai di Panduan AWS IAM Identity Center Pengguna. Untuk informasi selengkapnya tentang penyediaan akses pengguna Pusat Identitas IAM, lihat Akses masuk tunggal ke AWS akun di Panduan Pengguna. AWS IAM Identity Center

Gunakan kasus untuk IAM dengan Amazon EC2

Perusahaan seperti Example Corp biasanya menggunakan IAM untuk berinteraksi dengan layanan seperti Amazon EC2 Untuk memahami bagian kasus penggunaan ini, Anda memerlukan pemahaman dasar tentang Amazon EC2. Untuk informasi selengkapnya tentang Amazon EC2, lihat Panduan Pengguna Amazon EC2.

Izin Amazon EC2 untuk grup pengguna

Untuk menyediakan kontrol “perimeter”, Nikki melampirkan kebijakan ke grup AllUsers pengguna. Kebijakan ini menolak AWS permintaan apa pun dari pengguna jika alamat IP asal berada di luar jaringan perusahaan Example Corp.

Di Example Corp, grup IAM yang berbeda memerlukan izin yang berbeda:

- Administrator sistem - Perlu izin untuk membuat dan mengelola AMI, instans, snapshot, volume, grup keamanan, dan sebagainya. Nikki melampirkan `AmazonEC2FullAccess` AWS kebijakan terkelola ke grup SysAdmins pengguna yang memberikan izin kepada anggota grup untuk menggunakan semua tindakan Amazon EC2.
- Pengembang – Perlu kemampuan untuk bekerja hanya dengan instans. Oleh karena itu Mateo membuat dan melampirkan kebijakan ke grup pengguna Pengembang yang memungkinkan pengembang untuk memanggil`DescribeInstances`,`RunInstances`,, `StopInstances``StartInstances`, dan`TerminateInstances`.

  > **Note**
  >
  > Amazon EC2 menggunakan kunci SSH, kata sandi Windows, dan grup keamanan untuk mengontrol siapa yang memiliki akses ke sistem operasi dari instans Amazon EC2 tertentu. Tidak ada metode dalam sistem IAM untuk memungkinkan atau menolak akses ke sistem operasi dari suatu instans khusus.

- Mendukung pengguna — Seharusnya tidak dapat melakukan tindakan Amazon EC2 apa pun kecuali mencantumkan sumber daya Amazon EC2 yang saat ini tersedia. Oleh karena itu, Nikki membuat dan melampirkan kebijakan ke grup Pengguna Dukungan yang hanya memungkinkan mereka memanggil operasi API “Deskripsikan” Amazon EC2.

Untuk contoh seperti apa kebijakan ini, lihat Contoh kebijakan berbasis identitas IAM dan Menggunakan AWS Identitas dan Manajemen Akses di Panduan Pengguna Amazon EC2.

Perubahan fungsi pekerjaan pengguna

Pada titik tertentu, salah satu pengembang, Paulo Santos, mengubah fungsi pekerjaan dan menjadi manajer. Sebagai manajer, Paulo menjadi bagian dari grup pengguna Dukungan sehingga ia dapat membuka kasus dukungan untuk pengembangnya. Mateo memindahkan Paulo dari grup pengguna Pengembang ke grup pengguna Dukungan. Sebagai hasil dari langkah ini, kemampuannya untuk

berinteraksi dengan instans Amazon EC2 terbatas. Dia tidak dapat meluncurkan atau memulai instans. Dia juga tidak dapat menghentikan atau mematikan instans yang ada, bahkan jika dia pengguna yang meluncurkan atau memulai instans. Dia hanya dapat mencantumkan instans yang telah diluncurkan pengguna Example Corp.

Gunakan kasus untuk IAM dengan Amazon S3

Perusahaan seperti Example Corp biasanya menggunakan IAM dengan Amazon S3. John telah menciptakan bucket Amazon S3 untuk perusahaan yang disebut amz n-s3-demo-bucket.

Pembuatan pengguna dan grup pengguna lain

Sebagai karyawan, Zhang Wei dan Mary Major masing-masing harus dapat membuat data mereka sendiri di bucket perusahaan. Mereka juga perlu membaca dan menulis data yang dikerjakan oleh semua developer. Untuk mengaktifkan ini, Mateo secara logis mengatur data di amzn-s3-demo-bucket menggunakan skema awalan kunci Amazon S3 seperti yang ditunjukkan pada gambar berikut.

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

Mateo membaginya `/amzn-s3-demo-bucket` menjadi satu set direktori rumah untuk setiap karyawan, dan area bersama untuk kelompok pengembang dan manajer.

Sekarang Mateo membuat satu set kebijakan untuk menetapkan izin ke pengguna dan grup pengguna:

- Akses direktori beranda untuk Zhang — Mateo melampirkan kebijakan ke Wei yang memungkinkannya membaca, menulis, dan membuat daftar objek apa pun dengan awalan kunci Amazon S3 `/amzn-s3-demo-bucket/home/zhang/`
- Akses direktori beranda untuk Mayor — Mateo melampirkan kebijakan ke Mary yang memungkinkannya membaca, menulis, dan membuat daftar objek apa pun dengan awalan kunci Amazon S3 `/amzn-s3-demo-bucket/home/major/`

- Akses direktori bersama untuk grup pengguna pengembang - Mateo melampirkan kebijakan ke grup pengguna yang memungkinkan pengembang membaca, menulis, dan membuat daftar objek apa pun di `/amzn-s3-demo-bucket/share/developers/`
- Akses direktori bersama untuk grup pengguna manajer - Mateo melampirkan kebijakan ke grup pengguna yang memungkinkan manajer membaca, menulis, dan membuat daftar objek di `/amzn-s3-demo-bucket/share/managers/`

> **Note**
>
> Amazon S3 tidak secara otomatis memberi pengguna yang membuat izin bucket atau objek untuk melakukan tindakan lain pada bucket atau objek tersebut. Oleh karena itu, dalam kebijakan IAM, Anda harus secara eksplisit memberi pengguna izin untuk menggunakan sumber daya Amazon S3 yang mereka buat.

Untuk contoh tampilan kebijakan ini, lihat Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Untuk informasi tentang bagaimana kebijakan dievaluasi pada waktu aktif, lihat Logika evaluasi kebijakan.

Perubahan fungsi pekerjaan pengguna

Pada titik tertentu, salah satu pengembang, Zhang Wei, mengubah fungsi pekerjaan dan menjadi manajer. Kami berasumsi bahwa dia tidak lagi memerlukan akses ke dokumen dalam direktori `share/developers`. Mateo, sebagai admin, memindahkan Wei ke grup `Managers` pengguna dan keluar dari grup `Developers` pengguna. Hanya dengan penugasan ulang sederhana itu, Wei secara otomatis mendapatkan semua izin yang diberikan kepada grup `Managers` pengguna, tetapi tidak dapat lagi mengakses data di `share/developers` direktori.

Integrasi dengan bisnis pihak ketiga

Organisasi sering kali bekerja dengan perusahaan mitra, konsultan, dan kontraktor. Example Corp memiliki mitra yang disebut Widget Company, dan karyawan Perusahaan Widget bernama Shirley Rodriguez perlu memasukkan data ke dalam bucket untuk penggunaan Example Corp. Nikki membuat grup pengguna yang dipanggil WidgetCo dan pengguna bernama `Shirley` dan menambahkan Shirley ke grup WidgetCo pengguna. Nikki juga membuat ember khusus yang disebut amzn-s3- demo-bucket1 untuk digunakan Shirley.

Nikki memperbarui kebijakan yang ada atau menambahkan yang baru untuk mengakomodasi Perusahaan Widget mitra. Misalnya, Nikki dapat membuat kebijakan baru yang menolak kemampuan anggota grup WidgetCo pengguna untuk menggunakan tindakan apa pun selain menulis. Kebijakan ini hanya akan diperlukan jika terdapat kebijakan luas yang memberi semua pengguna akses ke serangkaian tindakan Amazon S3.

Tutorial IAM

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk tugas-tugas umum untuk AWS Identity and Access Management (IAM). Tutorial ini dimaksudkan untuk lingkungan tipe lab, dengan nama perusahaan fiktif , nama pengguna fiktif, dan sebagainya. Tujuannya adalah untuk memberikan pedoman umum. Produk ini tidak dimaksudkan untuk penggunaan langsung di lingkungan produksi tanpa tinjauan dan adaptasi cermat untuk memenuhi kebutuhan unik di lingkungan organisasi Anda.

Tutorial

- Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM
- Tutorial IAM: Buat dan lampirkan kebijakan yang dikelola pelanggan pertama Anda
- Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag
- Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka
- Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP)
- Tutorial IAM: Gunakan CloudFormation template untuk membuat peran IAM federasi SAMP
- Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP) dan peran IAM federasi SAML

Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM

> **Important**
>
> Praktik terbaik IAM menyarankan agar Anda mewajibkan pengguna manusia menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredentif sementara alih-alih menggunakan pengguna IAM dengan kredentif jangka panjang. Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

Tutorial ini mengajarkan Anda cara menggunakan peran untuk mendelegasikan akses ke sumber daya di berbagai yang Akun AWS disebut Tujuan dan Sumber. Anda berbagi sumber daya di satu akun dengan pengguna di akun yang berbeda. Dengan mengatur akses lintas akun dengan cara ini, Anda tidak perlu membuat pengguna IAM individual di setiap akun. Selain itu, pengguna tidak perlu

keluar dari satu akun dan masuk ke akun lain untuk mengakses sumber daya yang berbeda Akun AWS. Setelah mengonfigurasi peran, Anda melihat cara menggunakan peran dari Konsol Manajemen AWS, the AWS CLI, dan API.

Dalam tutorial ini, akun Tujuan mengelola data aplikasi yang diakses oleh aplikasi dan tim yang berbeda. Di setiap akun, Anda menyimpan informasi aplikasi di bucket Amazon S3. Anda mengelola pengguna IAM di akun Origining, di mana Anda memiliki dua peran pengguna IAM: Peng embang dan Analis. Pengembang dan Analis menggunakan akun Originating untuk menghasilkan data yang dibagikan oleh beberapa layanan mikro. Kedua peran memiliki izin untuk bekerja di akun Origin dan mengakses sumber daya di sana. Dari waktu ke waktu, pengembang harus memperbarui data bersama di akun Tujuan. Pengembang menyimpan data ini dalam bucket Amazon S3 yang disebut`amzn-s3-demo-bucket-shared-container`.

Di akhir tutorial ini, Anda memiliki hal berikut:

- Pengguna di akun Origin (akun tepercaya) diizinkan untuk mengambil peran tertentu di akun Tujuan.
- Peran di akun Tujuan (akun yang dipercaya) diizinkan untuk mengakses bucket Amazon S3 tertentu.
- `amzn-s3-demo-bucket-shared-container`Bucket di akun Tujuan.

Pengembang dapat menggunakan peran dalam Konsol Manajemen AWS untuk mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Mereka juga dapat mengakses bucket dengan menggunakan panggilan API yang diautentikasi oleh kredentif sementara yang disediakan oleh peran. Upaya serupa oleh Analis untuk menggunakan peran gagal.

Alur kerja ini memiliki tiga langkah dasar:

Buat peran di Akun Tujuan

Pertama, Anda menggunakan Konsol Manajemen AWS untuk membangun kepercayaan antara akun Tujuan (nomor ID 999999999999) dan akun Origin (nomor ID 11111111 1111). Anda mulai dengan membuat peran IAM bernama UpdateData. Saat Anda membuat peran, Anda menentukan akun Origin sebagai entitas tepercaya dan menentukan kebijakan izin yang memungkinkan pengguna tepercaya memperbarui `amzn-s3-demo-bucket-shared-container` bucket.

Berikan akses ke peran tersebut

Di bagian ini, Anda mengubah kebijakan peran untuk menolak akses Analis ke `UpdateData` peran. Karena Analis memiliki PowerUser akses dalam skenario ini, dan Anda harus secara eksplisit menolak kemampuan untuk menggunakan peran tersebut.

Akses uji dengan mengalihkan peran

Terakhir, sebagai Pengembang, Anda menggunakan `UpdateData` peran untuk memperbarui `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Anda melihat cara mengakses peran melalui AWS konsol, AWS CLI, dan API.

Pertimbangan-pertimbangan

Sebelum Anda menggunakan peran IAM untuk mendelegasikan akses sumber daya di seluruh dunia Akun AWS, penting untuk mempertimbangkan hal berikut:

- Anda tidak dapat beralih ke peran saat masuk sebagai Pengguna root akun AWS.
- Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dengan satu partisi. Misalnya, anggap saja Anda memiliki akun di AS Barat (California Utara) di partisi `aws` standar. Anda juga memiliki rekening di Tiongkok (Beijing) di partisi `aws-cn` tersebut. Anda tidak dapat menggunakan kebijakan berbasis sumber daya Amazon S3 di akun Anda di Tiongkok (Beijing) untuk mengizinkan akses bagi pengguna dalam akun standar `aws` Anda.
- Anda dapat menggunakannya AWS IAM Identity Center untuk memfasilitasi single sign-on (SSO) untuk eksternal Akun AWS (akun di luar Anda AWS Organizations) menggunakan Security Assertion Markup Language (SAML). Untuk detailnya, lihat Meng integrasikan eksternal Akun AWS ke dalam AWS IAM Identity Center untuk manajemen akses pusat dengan penagihan independen menggunakan SAML 2.0
- Anda dapat mengaitkan peran ke AWS sumber daya seperti instans atau AWS Lambda fungsi Amazon EC2. Lihat perinciannya di Buat peran untuk mendelegasikan izin ke AWS layanan.
- Jika Anda ingin aplikasi mengambil peran di aplikasi lain Akun AWS, Anda dapat menggunakan AWS SDK untuk asumsi peran lintas akun. Untuk informasi selengkapnya, lihat Otentikasi dan akses di Panduan Referensi AWS SDK dan Alat.
- Beralih peran menggunakan Konsol Manajemen AWS satu-satunya berfungsi dengan akun yang tidak memerlukan`ExternalId`. Misalnya, Anda memberikan akses untuk akun Anda ke pihak ketiga dan memerlukan `ExternalId` dalam elemen `Condition` dalam kebijakan izin Anda. Dalam hal ini, pihak ketiga dapat mengakses akun Anda hanya dengan menggunakan AWS API

atau alat baris perintah. Pihak ketiga tidak dapat menggunakan konsol karena harus memberikan nilai untuk`ExternalId`. Untuk informasi selengkapnya tentang skenario iniAkses ke Akun AWS dimiliki oleh pihak ketiga, lihat, dan Cara mengaktifkan akses lintas akun ke Konsol Manajemen AWS dalam Blog AWS Keamanan.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Dua terpisah Akun AWS yang dapat Anda gunakan, satu untuk mewakili akun Origin, dan satu untuk mewakili akun Tujuan.
- Pengguna dan peran dalam akun Origin dibuat dan dikonfigurasi sebagai berikut:

Judul pekerjaan	Pengguna	Perizinan
Developer	David	Kedua pengguna dapat masuk dan menggunakan akun Orig Konsol Manajemen AWS in.
Analis	Jane	

- Anda tidak perlu membuat pengguna apa pun di akun Tujuan.
- Bucket Amazon S3 yang dibuat di akun Tujuan. Anda dapat menyebut `amzn-s3-demo-bucket-shared-container` nya dalam tutorial ini, tetapi karena nama bucket S3 harus unik secara global, Anda harus menggunakan bucket dengan nama yang berbeda.

Buat peran di Akun Tujuan

Anda dapat mengizinkan pengguna dari satu Akun AWS untuk mengakses sumber daya di yang lain Akun AWS. Dalam tutorial ini, kita akan melakukan ini dengan membuat peran yang menentukan siapa yang dapat mengaksesnya dan izin apa yang diberikan kepada pengguna yang beralih ke itu.

Pada langkah tutorial ini, Anda membuat peran di akun Tujuan dan menentukan akun Origin sebagai entitas tepercaya. Anda juga membatasi izin peran untuk hanya membaca dan menulis akses ke `amzn-s3-demo-bucket-shared-container` bucket. Siapa pun yang diberi izin untuk menggunakan peran dapat membaca dan menulis ke `shared-container` bucket.

Sebelum Anda dapat membuat peran, Anda memerlukan ID akun dari Orig in Akun AWS. Masing-masing Akun AWS memiliki pengenal ID akun unik yang ditetapkan untuknya.

Untuk mendapatkan yang berasal Akun AWS ID

1. Masuk ke Konsol Manajemen AWS sebagai administrator akun Origin ating, dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di konsol IAM, pilih nama pengguna Anda di bilah navigasi di kanan atas. Biasanya terlihat seperti ini: ***username*** @ ***account_ID_number_or_alias***.

   Untuk skenario ini, Anda dapat menggunakan ID akun 111111111111 untuk akun Origin. Namun, Anda harus menggunakan ID akun yang valid jika Anda menggunakan skenario ini di lingkungan pengujian Anda.

Untuk membuat peran di akun Tujuan yang dapat digunakan oleh akun Origin

1. Masuk ke Konsol Manajemen AWS sebagai administrator akun Tujuan, dan buka konsol IAM.
2. Sebelum membuat peran, siapkan kebijakan terkelola yang mendefinisikan izin untuk persyaratan peran. Anda kemudian perlu melampirkan kebijakan ini ke peran tersebut pada langkah berikutnya.

   Anda ingin mengatur akses baca dan tulis ke bucket `amzn-s3-demo-bucket-shared-container`. Meskipun AWS menyediakan beberapa kebijakan terkelola Amazon S3, tidak ada yang menyediakan akses baca dan tulis ke satu bucket Amazon S3. Anda dapat membuat kebijakan Anda sendiri.

   Pada panel navigasi, silakan pilih Kebijakan dan kemudian pilih Buat kebijakan.
3. Pilih tab JSON dan salin teks dari dokumen kebijakan JSON berikut. Tempel teks ini ke dalam kotak teks JSON, ganti sumber daya ARN (`arn:aws:s3:::shared-container`) dengan yang asli untuk bucket Amazon S3 Anda.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
```

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

T `ListAllMyBuckets` indakan memberikan izin untuk mencantumkan semua bucket yang dimiliki oleh pengirim permintaan yang diautentikasi. Izin `ListBucket` memungkinkan pengguna melihat objek dalam bucket `amzn-s3-demo-bucket-shared-container`. Izin `GetObject`, `PutObject`, `DeleteObject` memungkinkan pengguna melihat, memperbarui, dan menghapus konten di bucket `amzn-s3-demo-bucket-shared-container`.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

4. Pada halaman Tinjau dan buat, **`read-write-app-bucket`** ketik nama kebijakan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola.

5. Pada panel navigasi, silakan pilih Peran lalu pilih Buat peran.

6. Pilih jenis Akun AWS peran An.
7. Untuk ID Akun, ketik ID akun asal.

   Tutorial ini menggunakan contoh ID akun **111111111111** untuk akun Origin. Anda harus menggunakan ID akun yang valid. Jika Anda menggunakan ID akun yang tidak valid, seperti **111111111111**, IAM tidak mengizinkan Anda membuat peran baru.

   Untuk saat ini, Anda tidak perlu memerlukan ID eksternal, atau mewajibkan pengguna untuk memiliki Multi-Factor Authentication (MFA) untuk menjalankan peran tersebut. Biarkan opsi ini tidak dipilih. Untuk informasi selengkapnya, lihat AWS Multi-factor otentikasi di IAM.
8. Pilih Berikutnya: Izin untuk mengatur izin yang terkait dengan peran.
9. Pilih kotak centang di sebelah kebijakan yang Anda buat sebelumnya.

   > **Kiat**
   >
   > Untuk Filter, pilih Pelang gan berhasil memfilter daftar agar hanya menyertakan kebijakan yang Anda buat. Ini menyembunyikan kebijakan yang AWS dibuat dan membuatnya lebih mudah untuk menemukan yang Anda butuhkan.

   Lalu, pilih Selanjutnya.
10. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
11. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
12. Setelah meninjau peran, klik Buat peran.

    Peran `UpdateData` muncul di daftar peran.

Sekarang Anda harus mendapatkan Nama Sumber Daya Amazon (ARN) dari peran tersebut, pengidentifikasi unik untuk peran tersebut. Saat Anda mengubah peran Pengembang di akun Origin, Anda menentukan peran ARN dari akun Tujuan untuk memberikan atau menolak izin.

Untuk mendapatkan ARN untuk UpdateData

1. Di panel navigasi konsol IAM, pilih Peran.

2. Dalam daftar peran, pilih peran `UpdateData`.
3. Di bagian Ringkasan di panel perincian, salin nilai ARN Peran.

   Akun Tujuan memiliki ID akun 999999999999, jadi peran ARN adalah. `arn:aws:iam::999999999999:role/UpdateData` Pastikan Anda memberikan Akun AWS ID asli untuk akun Tujuan.

Pada titik ini, Anda telah membangun kepercayaan antara akun Tujuan dan Asal. Anda melakukan ini dengan membuat peran di akun Tujuan yang mengidentifikasi akun A sal sebagai prinsipal tepercaya. Anda juga menentukan apa yang dapat dilakukan oleh pengguna yang beralih ke `UpdateData` peran.

Selanjutnya, ubah izin untuk peran Pengembang.

Berikan akses ke peran tersebut

Pada titik ini, Analis dan Pengembang memiliki izin yang memungkinkan mereka mengelola data di akun Origin. Gunakan langkah-langkah berikut yang diperlukan untuk menambahkan izin untuk memungkinkan beralih ke peran.

Untuk memodifikasi peran Pengembang untuk memungkinkan mereka beralih ke UpdateData peran

1. Masuk sebagai administrator di akun Originating, dan buka konsol IAM.
2. Pilih Per an, lalu pilih Peng embang.
3. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
4. Pilih tab JSON.
5. Tambahkan pernyataan kebijakan berikut untuk mengiz `AssumeRole` inkan tindakan pada `UpdateData` peran di akun Tujuan. Pastikan bahwa Anda mengubah *`DESTINATION-ACCOUNT-ID`* `Resource` elemen ke Akun AWS ID aktual dari akun Tujuan.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
```

```
        "Resource": "arn:aws:iam::111122223333:role/UpdateData"
    }
}
```

`Allow`Efek ini secara eksplisit memungkinkan akses grup Pengembang ke `UpdateData` peran di akun Tujuan. Setiap pengembang yang mencoba mengakses peran berhasil.

6. Pilih Tinjau kebijakan.
7. Ketik Nama seperti**`allow-assume-S3-role-in-destination`**.
8. Pilih Buat kebijakan.

Di sebagian besar lingkungan, Anda mungkin tidak memerlukan prosedur berikut. Namun, jika Anda menggunakan PowerUserAccess izin, maka beberapa grup mungkin sudah dapat beralih peran. Prosedur berikut menunjukkan cara menambahkan `"Deny"` izin ke grup Analis untuk memastikan bahwa mereka tidak dapat mengambil peran tersebut. Jika Anda tidak memerlukan prosedur ini di lingkungan Anda, maka kami sarankan Anda tidak menambahkannya. `"Deny"`izin membuat gambaran izin keseluruhan lebih rumit untuk dikelola dan dipahami. Gunakan `"Deny"` izin hanya jika Anda tidak memiliki opsi yang lebih baik.

Untuk memodifikasi peran Analis untuk menolak izin untuk mengambil **`UpdateData`** peran

1. Pilih Per an, lalu pilih Analis.
2. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
3. Pilih tab JSON.
4. Tambahkan pernyataan kebijakan berikut untuk menolak tindakan `AssumeRole` di peran `UpdateData`. Pastikan bahwa Anda mengubah *`DESTINATION-ACCOUNT-ID`* `Resource` elemen ke Akun AWS ID aktual dari akun Tujuan.

   JSON

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

```
    }
}
```

`Deny`Efek secara eksplisit menolak akses grup Analis ke `UpdateData` peran di akun Tujuan. Setiap analis yang mencoba mengakses peran menerima pesan akses ditolak.

5. Pilih Tinjau kebijakan.
6. Ketik Nama seperti**`deny-assume-S3-role-in-destination`**.
7. Pilih Buat kebijakan.

Peran Pengembang sekarang memiliki izin untuk menggunakan `UpdateData` peran di akun Tujuan. Peran Analis dicegah untuk menggunakan `UpdateData` peran tersebut.

Selanjutnya, Anda dapat melihat bagaimana David, seorang pengembang, dapat mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. David dapat mengakses bucket dari Konsol Manajemen AWS, the AWS CLI, atau AWS API.

Akses uji dengan mengalihkan peran

Setelah menyelesaikan dua langkah pertama dari tutorial ini, Anda memiliki peran yang memberikan akses ke sumber daya di akun Tujuan. Anda juga memiliki satu peran di akun Origin dengan pengguna yang diizinkan untuk menggunakan peran itu. Langkah ini membahas cara menguji pengalihan ke peran itu dari Konsol Manajemen AWS AWS CLI,, dan AWS API.

Untuk mendapatkan bantuan dengan masalah umum yang mungkin Anda temui saat bekerja dengan peran IAM, lihatMemecahkan masalah peran IAM.

Beralih peran (konsol)

Jika David perlu memperbarui data di akun Tujuan di Konsol Manajemen AWS, dia dapat melakukannya dengan menggunakan Peralihan Peran. Dia menentukan ID akun atau alias dan nama peran, dan izinnya segera dialihkan kepada mereka yang diizinkan oleh peran tersebut. Dia kemudian dapat menggunakan konsol untuk bekerja dengan `amzn-s3-demo-bucket-shared-container` bucket, tetapi tidak dapat bekerja dengan sumber daya lain di Dest ination. Sementara David menggunakan peran tersebut, dia juga tidak dapat menggunakan hak istimewa pengguna kekuatannya di akun Orig ining. Itu karena hanya satu set izin yang dapat berlaku pada satu waktu.

IAM menyediakan dua cara yang dapat digunakan David untuk masuk ke halaman Peral ihan Peran:

- David menerima tautan dari administrator mereka yang menunjuk ke konfigurasi Peran Switch yang telah ditentukan sebelumnya. Tautan ini diberikan ke administrator di halaman terakhir dari panduan Buat peran atau pada halaman Ringkasan Peran untuk peran lintas akun. Memilih tautan ini membawa David ke halaman Alihkan Peran dengan bidang ID Akun dan Nama peran yang sudah diisi. Yang perlu dilakukan David adalah memilih Switch Roles.
- Administrator tidak mengirim tautan di surel, tetapi mengirim nilai ID Akun dan Nama Peran. Untuk beralih peran, David harus memasukkan nilai secara manual. Hal ini digambarkan dalam prosedur berikut.

Untuk mengambil peran

1. David masuk Konsol Manajemen AWS menggunakan pengguna normalnya di akun Origining.
2. Mereka memilih tautan yang dikirim oleh administrator melalui email kepada mereka. Ini membawa David ke halaman Switch Role dengan ID akun atau alias dan informasi nama peran yang sudah diisi.

   — atau —

   David memilih nama mereka (menu Identitas) di bilah navigasi, dan kemudian memilih Ber alih Peran.

   Jika ini adalah pertama kalinya David mencoba mengakses halaman Switch Role dengan cara ini, dia pertama kali mendarat di halaman Switch Role yang dijalankan pertama kali. Halaman ini memberikan informasi tambahan tentang bagaimana pengalihan peran dapat memungkinkan pengguna mengelola sumber daya di seluruh Akun AWS. David harus memilih Beralih Peran di halaman ini untuk menyelesaikan sisa prosedur ini.
3. Selanjutnya, untuk mengakses peran, David harus secara manual mengetikkan nomor ID akun Tujuan (999999999999) dan nama peran (`UpdateData`).

   Selain itu, David ingin memantau peran dan izin terkait yang saat ini aktif di IAM. Untuk melacak informasi ini, dia mengetik `Destination` di kotak teks Display Nama (Nama Tampilan), memilih opsi warna merah, lalu memilih Switch Role (Alihkan Peran).
4. Sekarang, David dapat menggunakan konsol Amazon S3 untuk bekerja menggunakan bucket Amazon S3 atau sumber daya lain yang peran `UpdateData`nya memiliki izin.
5. Setelah selesai, David dapat kembali ke izin aslinya. Untuk melakukan itu, mereka memilih nama tampilan peran Tujuan pada bilah navigasi dan kemudian memilih Kem bali ke David @ 111111111111.

6. Lain kali David ingin beralih peran dan memilih menu Identitas di bilah navigasi, dia melihat entri Tujuan masih ada dari terakhir kali. Dia cukup memilih entri itu untuk beralih peran segera tanpa memasukkan kembali ID akun dan nama peran.

Beralih peran (AWS CLI)

Jika David perlu bekerja di lingkungan Tujuan di baris perintah, dia dapat melakukannya dengan menggunakan AWS CLI. Dia menjalankan perintah `aws sts assume-role` dan meneruskan peran ARN untuk mendapatkan kredensial keamanan sementara untuk peran tersebut. Dia kemudian mengonfigurasi kredenSIAL tersebut dalam variabel lingkungan sehingga AWS CLI perintah berikutnya berfungsi menggunakan izin peran. Sementara David menggunakan peran tersebut, dia tidak dapat menggunakan hak istimewa pengguna kekuatannya di akun Origining, karena hanya satu set izin yang dapat berlaku pada satu waktu.

Perhatikan bahwa semua access key dan token hanyalah contoh dan tidak dapat digunakan seperti yang ditunjukkan. Ganti dengan nilai yang sesuai dari lingkungan langsung Anda.

Untuk mengambil peran

1. David membuka jendela command prompt, dan mengonfirmasi bahwa AWS CLI klien bekerja dengan menjalankan perintah:

   ```
   aws help
   ```

   > **Note**
   >
   > Lingkungan default David menggunakan kredensial pengguna `David` dari profil default yang ia buat dengan perintah `aws configure`. Untuk informasi lebih lanjut, lihat Mengonfigurasi AWS Command Line Interface di Panduan Pengguna AWS Command Line Interface .

2. Dia memulai proses peralihan peran dengan menjalankan perintah berikut untuk beralih ke `UpdateData` peran di akun Tujuan. Dia menerima peran ARN dari administrator yang menciptakan peran tersebut. Perintah ini mengharuskan Anda memberikan nama sesi, Anda juga dapat memilih teks apa pun yang Anda suka untuk itu.

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

David kemudian melihat hal-hal berikut di output:

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

3. David melihat tiga bagian yang mereka butuhkan di bagian Credentials dari output.

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

   David perlu mengkonfigurasi AWS CLI lingkungan untuk menggunakan parameter ini dalam panggilan berikutnya. Untuk informasi tentang berbagai cara mengonfigurasi kredensial Anda, lihat Mengonfigurasi AWS Command Line Interface. Anda tidak dapat menggunakan perintah `aws configure` karena ia tindak mendukung perekaman token sesi. Namun, Anda dapat memasukkan informasi secara manual ke dalam file konfigurasi. Karena ini adalah kredensial sementara dengan waktu kedaluwarsa yang relatif singkat, paling mudah menambahkannya ke lingkungan sesi baris perintah Anda saat ini.

4. Untuk menambahkan tiga nilai ke lingkungan, David memotong dan menempelkan output langkah sebelumnya ke perintah berikutnya. Anda mungkin ingin memotong dan menempel ke dalam editor teks sederhana untuk mengatasi masalah line wrap dalam output token sesi. Ia harus ditambahkan sebagai satu string panjang, meskipun line wrap ditampilkan di sini untuk kejelasan.

Contoh berikut menunjukkan perintah yang diberikan di lingkungan Windows, di mana “set” adalah perintah untuk membuat variabel lingkungan. Di komputer Linux atau macOS, Anda akan menggunakan perintah “ekspor” sebagai gantinya. Semua bagian contoh lainnya valid di ketiga lingkungan.

Untuk detail tentang menggunakan Tools untuk Windows Powershell, lihat Beralih ke peran IAM (Alat untuk Windows PowerShell)

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

Pada titik ini, setiap perintah berikut dijalankan di bawah izin peran yang diidentifikasi oleh kredensial tersebut. Dalam kasus David, peran `UpdateData`.

> **Important**
>
> Anda dapat menyimpan pengaturan konfigurasi dan kredenSIAL yang sering digunakan dalam file yang dikelola oleh AWS CLI. Untuk informasi selengkapnya, lihat Menggunakan file konfigurasi dan kredenSIAL yang ada di Panduan AWS Command Line Interface Pengguna.

5. Jalankan perintah untuk mengakses sumber daya di akun Tujuan. Dalam contoh ini, David mencantumkan isi bucket S3 mereka dengan perintah berikut.

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

Karena nama bucket Amazon S3 unik secara universal, tidak perlu menentukan ID akun yang memiliki bucket. Untuk mengakses sumber daya untuk AWS layanan lain, lihat AWS

CLI dokumentasi untuk layanan tersebut untuk perintah dan sintaks yang diperlukan untuk mereferensikan sumber dayanya.

Menggunakan AssumeRole (AWS API)

Ketika David perlu membuat pembaruan ke akun Tujuan dari kode, dia membuat `AssumeRole` panggilan untuk mengambil `UpdateData` peran tersebut. Panggilan mengembalikan kredenSIAL sementara yang dapat ia gunakan untuk mengakses `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan. Dengan kredensial tersebut, David dapat melakukan panggilan API untuk memperbarui bucket `amzn-s3-demo-bucket-shared-container`. Namun, ia tidak dapat melakukan panggilan API untuk mengakses sumber daya lain di akun Tujuan, meskipun ia memiliki izin pengguna kuat di akun Orig in.

Untuk mengambil peran

1. David memanggil `AssumeRole` sebagai bagian dari aplikasi. Mereka harus menentukan AR `UpdateData` N:`arn:aws:iam::999999999999:role/UpdateData`.

   Respond dari panggilan `AssumeRole` mencakup kredensial sementara dengan `AccessKeyId` dan `SecretAccessKey`. Informasi ini juga mencakup waktu `Expiration` yang menunjukkan kapan kredensial kedaluwarsa dan Anda harus meminta yang baru. Saat Anda mengatur rantai peran dengan AWS SDK, banyak penyedia kredensia secara otomatis menyegarkan kredensi sebelum kedaluwarsa.
2. Dengan kredensial sementara, David membuat panggilan `s3:PutObject` untuk memperbarui bucket `amzn-s3-demo-bucket-shared-container`. Mereka akan meneruskan kredenSIAL ke panggilan API sebagai `AuthParams` parameter. Karena kredentif peran sementara hanya memiliki akses baca dan tulis ke `amzn-s3-demo-bucket-shared-container` bucket, tindakan lain di akun Tujuan ditolak.

Untuk contoh kode (menggunakan Python), lihat Beralih ke peran IAM (AWS API).

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang topik dalam tutorial ini:

- Untuk informasi selengkapnya tentang pengguna IAM, lihatIdentitas IAM.
- Untuk informasi selengkapnya tentang bucket Amazon S3, lihat Membuat Bucket di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

- Untuk mempelajari apakah prinsip di akun di luar zona kepercayaan (organisasi atau akun tepercaya) memiliki akses untuk menjalankan peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

Ringkasan

Anda telah menyelesaikan tutorial akses API lintas akun. Anda menciptakan peran untuk membangun kepercayaan dengan akun lainnya dan menetapkan tindakan apa yang dapat dilakukan oleh entitas tepercaya. Kemudian, Anda mengubah kebijakan peran untuk mengontrol pengguna IAM mana yang dapat mengakses peran tersebut. Akibatnya, pengembang dari akun Orig in dapat melakukan pembaruan ke `amzn-s3-demo-bucket-shared-container` bucket di akun Tujuan dengan menggunakan kredenSIAL sementara.

Tutorial IAM: Buat dan lampirkan kebijakan yang dikelola pelanggan pertama Anda

Dalam tutorial ini, Anda menggunakan Konsol Manajemen AWS untuk membuat kebijakan yang dikelola pelanggan dan kemudian melampirkan kebijakan itu ke pengguna IAM di Anda Akun AWS. Kebijakan yang Anda buat memungkinkan pengguna uji IAM untuk masuk langsung ke Konsol Manajemen AWS dengan izin hanya-baca.

Alur kerja ini memiliki tiga langkah dasar:

Langkah 1: Buat kebijakan

Secara default, pengguna IAM tidak memiliki izin untuk melakukan apa pun. Mereka tidak dapat mengakses AWS Management Console atau mengelola data di dalamnya, kecuali jika Anda mengizinkannya. Pada langkah ini, Anda membuat kebijakan yerkelola pelanggan yang memungkinkan setiap pengguna yang melekat untuk masuk ke konsol.

Langkah 2: Lampirkan kebijakan

Saat Anda melampirkan kebijakan kepada pengguna, pengguna menurunkan semua izin akses yang terkait dengan kebijakan tersebut. Pada langkah ini, Anda melampirkan kebijakan baru ke pengguna pengujian.

Langkah 3: Uji akses pengguna

Setelah kebijakan tersebut dilampirkan, Anda dapat masuk sebagai pengguna dan menguji kebijakan tersebut.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS yang dapat Anda masuk sebagai pengguna IAM dengan izin administratif.
- Pengguna IAM uji yang tidak memiliki izin yang diberikan atau keanggotaan grup sebagai berikut:

Nama pengguna	Grup	Izin
PolicyUser	<tidak ada>	<tidak ada>

Langkah 1: Buat kebijakan

Pada langkah ini, Anda membuat kebijakan terkelola pelanggan yang memungkinkan pengguna terlampir masuk Konsol Manajemen AWS dengan akses hanya-baca ke data IAM.

Untuk membuat kebijakan bagi pengguna uji Anda

1. Masuk ke konsol IAM di https://console.aws.amazon.com/iam/dengan pengguna Anda yang memiliki izin administrator.
2. Di panel navigasi, pilih Kebijakan.
3. Di panel konten, pilih Buat kebijakan.
4. Pilih opsi JSON dan salin teks dari dokumen kebijakan JSON berikut. Tempel teks ini ke kotak teks JSON.

   JSON

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

```
    } ]
}
```

5. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda membuat perubahan atau memilih Tinjau kebijakan di Editor visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

6. Pada halaman Tinjau dan buat, ketik **`UsersReadOnlyAccessToIAMConsole`** nama kebijakan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola dan siap dilampirkan.

Langkah 2: Lampirkan kebijakan

Selanjutnya, Anda melampirkan kebijakan yang baru saja Anda buat ke pengguna IAM uji Anda.

Untuk melampirkan kebijakan ke pengguna uji Anda

1. Di konsol IAM, di panel navigasi, pilih Kebijakan.
2. Di bagian atas daftar kebijakan, di kotak pencarian, mulailah mengetik **`UsersReadOnlyAccessToIAMConsole`** hingga Anda dapat melihat kebijakan Anda. Kemudian pilih tombol radio UsersReadOnlyAccessToIAMConsoledi sebelah dalam daftar.
3. Pilih tombol Tindakan, lalu pilih Lampirkan.
4. Dalam entitas IAM pilih opsi untuk memfilter untuk Pengguna.
5. Di kotak pencarian, mulailah mengetik **`PolicyUser`** hingga pengguna terlihat di daftar. Lalu centang kotak di samping pengguna tersebut di daftar.
6. Pilih Lampirkan kebijakan.

Anda telah melampirkan kebijakan tersebut ke pengguna uji IAM Anda, yang berarti bahwa pengguna sekarang memiliki akses hanya-baca ke konsol IAM.

Langkah 3: Uji akses pengguna

Untuk tutorial ini, kami sarankan agar Anda menguji akses dengan masuk sebagai pengguna uji sehingga Anda dapat melihat apa yang pengguna mungkin alami.

Untuk menguji akses dengan masuk dengan pengguna pengujian

1. Masuk ke konsol IAM di https://console.aws.amazon.com/iam/dengan pengguna `PolicyUser` pengujian Anda.
2. Jelajahi halaman konsol dan coba buat pengguna atau grup baru. Perhatikan bahwa `PolicyUser` dapat menampilkan data tetapi tidak dapat membuat atau memodifikasi data IAM yang ada.

Sumber daya terkait

Untuk informasi terkait, lihat sumber daya berikut:

- Kebijakan terkelola dan kebijakan inline
- Kontrol akses pengguna IAM ke Konsol Manajemen AWS

Ringkasan

Anda telah berhasil menyelesaikan semua langkah yang diperlukan untuk membuat dan melampirkan kebijakan terkelola pelanggan. Oleh karena itu, Anda dapat masuk ke konsol IAM dengan akun uji Anda untuk melihat seperti apa pengalaman bagi pengguna Anda.

Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag

Attribute-based kontrol akses (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Di AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke sumber daya IAM, termasuk entitas IAM (pengguna atau peran) dan ke AWS sumber daya. Kemudian, Anda dapat menentukan kebijakan yang menggunakan kunci syarat tanda untuk memberikan izin kepada penanggung jawab Anda berdasarkan tanda mereka. Saat Anda menggunakan tag untuk mengontrol akses ke AWS sumber daya, Anda memungkinkan tim dan sumber daya tumbuh dengan sedikit perubahan AWS kebijakan. Kebijakan ABAC lebih fleksibel daripada AWS kebijakan tradisional, yang mengharuskan

Anda mencantumkan setiap sumber daya individu. Untuk informasi selengkapnya tentang ABAC dan manfaatnya dibandingkan kebijakan tradisional, lihat Tentukan izin berdasarkan atribut dengan otorisasi ABAC.

> **Note**
>
> Anda harus meneruskan satu nilai untuk setiap tag sesi. AWS Security Token Service tidak mendukung tag sesi multi-nilai.

Topik

- Ikhtisar tutorial
- Prasyarat
- Langkah 1: Buat pengguna uji
- Langkah 2: Buat kebijakan ABAC
- Langkah 3: Buat peran
- Langkah 4: Uji pembuatan rahasia
- Langkah 5: Uji menampilkan rahasia
- Langkah 6: Uji skalabilitas
- Langkah 7: Uji memperbarui dan menghapus rahasia
- Ringkasan
- Sumber daya terkait
- Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC

Ikhtisar tutorial

Tutorial ini menunjukkan cara membuat dan menguji sebuah kebijakan yang memungkinkan peran IAM dengan tag penanggung jawab untuk mengakses sumberdaya dengan tag yang sesuai. Saat prinsipal mengajukan permintaan ke AWS, izin mereka diberikan berdasarkan apakah tanda prinsipal dan tanda sumber daya cocok. Strategi ini memungkinkan individu untuk melihat atau mengedit hanya AWS sumber daya yang diperlukan untuk pekerjaan mereka.

Skenario

Asumsikan bahwa Anda adalah pengembang utama di perusahaan besar bernama Perusahaan Contoh, dan Anda adalah administrator IAM berpengalaman. Anda akrab dengan membuat dan mengelola pengguna, peran, dan kebijakan IAM. Anda ingin memastikan bahwa anggota teknisi pengembangan dan tim jaminan kualitas Anda dapat mengakses sumber daya yang mereka butuhkan. Anda juga memerlukan strategi yang dapat menskalakan pertumbuhan perusahaan Anda.

Anda memilih untuk menggunakan tag AWS sumber daya dan tag utama peran IAM untuk menerapkan strategi ABAC untuk layanan yang mendukungnya, dimulai dengan. AWS Secrets Manager Untuk mempelajari layanan mana yang mendukung otorisasi berdasarkan tanda, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari kunci kondisi penandaan mana yang dapat Anda gunakan dalam kebijakan dengan tindakan dan sumber daya masing-masing layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan. Anda dapat mengonfigurasi penyedia identitas Anda SAML-based atau web untuk mener uskan tag sesi ke AWS. Ketika karyawan Anda bergabung AWS, atribut mereka diterapkan pada prinsipal yang dihasilkan. AWS Kemudian, Anda dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut. Untuk mempelajari cara menggunakan tanda sesi dengan identitas gabungan SAML yang berbeda dengan tutorial ini, lihat Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC.

Anggota tim Insinyur dan Jaminan Kualitas Anda berada di proyek Pegasus atau proyekUnicorn Anda. Anda memilih proyek 3 karakter dan nilai tanda tim berikut:

- `access-project` = `peg` untuk proyek Pegasus
- `access-project` = `uni` untuk proyek Unicorn
- `access-team` = `eng` untuk tim Insinyur
- `access-team` = `qas` untuk tim Jaminan Kualitas

Selain itu, Anda memilih untuk mewajibkan tag alokasi `cost-center` biaya untuk mengaktifkan laporan AWS penagihan kustom. Untuk informasi selengkapnya, lihat Penggunaan Tanda Alokasi Biaya dalam Panduan Pengguna AWS Manajemen Penagihan dan Biaya .

Ringkasan keputusan penting

- Karyawan masuk dengan kredensial pengguna IAM dan kemudian mengambil peran IAM untuk tim dan proyek mereka. Jika perusahaan Anda memiliki sistem identitas tersendiri, Anda dapat mengatur federasi untuk memungkinkan karyawan mengambil peran tanpa pengguna IAM. Untuk informasi selengkapnya, lihat Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC.

- Kebijakan yang sama terlampir pada semua peran. Tindakan diizinkan atau ditolak berdasarkan tanda.
- Karyawan dapat membuat sumber daya baru, tetapi hanya jika mereka melampirkan tanda yang sama ke sumber daya yang diterapkan pada peran mereka. Hal ini memastikan bahwa karyawan dapat melihat sumber daya setelah mereka membuatnya. Administrator tidak lagi diwajibkan untuk memperbarui kebijakan dengan ARN dari sumber daya baru.
- Karyawan dapat membaca sumber daya yang dimiliki oleh tim mereka, apa pun proyeknya.
- Karyawan dapat memperbarui dan menghapus sumber daya yang dimiliki oleh tim dan proyek mereka sendiri.
- Administrator IAM dapat menambahkan peran baru untuk proyek baru. Mereka dapat membuat dan menandai pengguna IAM baru untuk memungkinkan akses ke peran yang sesuai. Administrator tidak diwajibkan untuk mengedit kebijakan untuk mendukung proyek atau anggota tim baru.

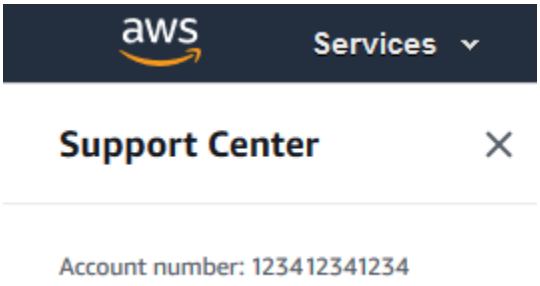
Di tutorial ini, Anda akan menandai setiap sumber daya, menandai peran proyek Anda, dan menambahkan kebijakan ke peran untuk memungkinkan perilaku yang dijelaskan sebelumnya. Kebijakan yang dihasilkan memungkinkan peran `Create`, `Read`, `Update`, dan `Delete` mengakses sumber daya yang ditandai dengan tanda proyek dan tanda tim yang sama. Kebijakan ini juga memungkinkan proyek silang `Read` untuk mengakses sumber daya yang ditandai dengan tim yang sama.
Diagram menunjukkan dua proyek di mana peran terbatas untuk akses baca saja di luar proyek mereka sementara memiliki izin untuk membuat, membaca, memperbarui, dan menghapus sumber daya dalam proyek mereka sendiri.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS tempat Anda dapat masuk sebagai pengguna dengan izin administratif.
- ID akun 12 digit Anda, yang Anda gunakan untuk membuat peran di langkah 3.

  Untuk menemukan nomor ID AWS akun Anda menggunakan Konsol Manajemen AWS, pilih Dukungan pada bilah navigasi di kanan atas, lalu pilih Pusat Dukungan. Nomor akun (ID) muncul di panel navigasi di sebelah kiri.



- Pengalaman membuat dan mengedit pengguna, peran, dan kebijakan IAM di Konsol Manajemen AWS. Tetapi, jika Anda memerlukan bantuan untuk mengingat proses manajemen IAM, tutorial ini menyediakan tautan di mana Anda dapat melihat petunjuk langkah demi langkah.

Langkah 1: Buat pengguna uji

Untuk pengujian, buat empat pengguna IAM dengan izin untuk menerima peran dengan tag yang sama. Ini memudahkan untuk menambahkan lebih banyak pengguna ke tim Anda. Saat Anda menandai pengguna, mereka secara otomatis mendapatkan akses untuk mengambil peran yang tepat. Anda tidak perlu menambahkan pengguna ke kebijakan kepercayaan peran tersebut jika mereka hanya menangani satu proyek dan tim.

1. Buat kebijakan yang dikelola pelanggan berikut bernama `access-assume-role`. Untuk informasi selengkapnya tentang membuat kebijakan JSON, lihat Membuat kebijakan IAM.

   Kebijakan ABAC: Asumsikan peran ABAC apa pun, tetapi hanya ketika peran pengguna cocok dengan tanda peran

   Kebijakan berikut memungkinkan pengguna mengambil peran apa pun di akun Anda dengan prefiks nama `access-`. Peran tersebut juga harus ditandai dengan tanda proyek, tim, dan pusat biaya yang sama dengan pengguna.

   Untuk menggunakan kebijakan ini, ganti teks plac eholder miring dengan informasi akun Anda.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
```

```
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111122223333:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/
access-project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/
access-team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Untuk menskalakan tutorial ini ke sejumlah besar pengguna, Anda dapat melampirkan kebijakan ke grup dan menambahkan setiap pengguna ke grup. Untuk informasi selengkapnya, lihat Buat grup IAM dan Mengedit pengguna dalam grup IAM.

2. Buat pengguna IAM berikut, lampirkan kebijakan `access-assume-role` izin. Pastikan Anda memilih Berikan akses pengguna ke Konsol Manajemen AWS, lalu tambahkan tag berikut.

Nama pengguna	Kunci tag pengguna	Nilai tag pengguna
akses- Arnav-peg-eng	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
akses- Mary-peg-qas	`access-project`	`peg`
	`access-team`	`qas`
	`cost-center`	`987654`
akses- Saanvi-uni-eng	`access-project`	`uni`
	`access-team`	`eng`

Nama pengguna	Kunci tag pengguna	Nilai tag pengguna
	`cost-center`	`123456`

akses- Carlos-uni-qas	`access-project`
	
	`access-team`
	
	`cost-center`
		`uni`
		
		`qas`
		
		`123456`

Langkah 2: Buat kebijakan ABAC

Buat kebijakan berikut bernama **`access-same-project-team`**. Anda akan menambahkan kebijakan ini ke peran tersebut pada langkah berikutnya. Untuk informasi selengkapnya tentang membuat kebijakan JSON, lihat Membuat kebijakan IAM.

Untuk kebijakan tambahan yang dapat Anda sesuaikan untuk tutorial ini, lihat halaman berikut:

- Mengontrol akses untuk prinsipal IAM
- Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol
- EC2: Mulai atau hentikan instans berdasarkan pada pencocokan prinsipal dan tanda sumber daya
- EC2: Mulai atau hentikan instance berdasarkan tag
- IAM: Asumsikan peran yang memiliki tag tertentu

Kebijakan ABAC: Akses Sumber Daya Manajer Rahasia Hanya Saat Tag Principal dan Sumber Daya Cocokkan

Kebijakan berikut memungkinkan penanggung jawab membuat, membaca, mengedit, dan menghapus sumber daya, tetapi hanya jika sumber daya tersebut ditandai dengan pasangan nilai kunci yang sama dengan penanggung jawab. Saat prinsipal membuat sumber daya, mereka harus menambahkan tanda `access-project`, `access-team`, dan `cost-center` dengan nilai yang cocok dengan tanda prinsipal. Kebijakan juga memungkinkan penambahan opsi tanda `Name` atau `OwnedBy`.

JSON

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

```
    },
    {
        "Sid": "ReadSecretsManagerSameTeam",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:Describe*",
            "secretsmanager:Get*",
            "secretsmanager:List*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
            }
        }
    },
    {
        "Sid": "DenyUntagSecretsManagerReservedTags",
        "Effect": "Deny",
        "Action": "secretsmanager:UntagResource",
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringLike": {
                "aws:TagKeys": "access-*"
            }
        }
    },
    {
        "Sid": "DenyPermissionsManagement",
        "Effect": "Deny",
        "Action": "secretsmanager:*Policy",
        "Resource": "*"
    }
 ]
}
```

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllActionsSecretsManagerSameProjectSameTeam` memungkinkan semua tindakan layanan ini pada semua sumber daya terkait, tetapi hanya jika tanda sumber daya cocok dengan tanda prinsipal. Dengan menambahkan `"Action": "secretsmanager:*"` ke kebijakan, akan menumbuhkan kebijakan seiring bertumbuhnya Secrets Manager. Jika Secrets

Manager menambahkan operasi API baru, Anda tidak perlu menambahkan tindakan tersebut ke pernyataan. Pernyataan tersebut menerapkan ABAC menggunakan tiga blok kondisi. Permintaan hanya diperbolehkan jika ketiga blok dikembalikan dengan benar.

- Blok kondisi pertama dari pernyataan ini dikembalikan dengan benar jika kunci tanda yang ditentukan ada pada sumber daya, dan nilainya cocok dengan tanda prinsipal. Blok ini mengembalikan kesalahan tanda yang tidak sesuai, atau tindakan yang tidak mendukung penandaan sumber daya. Untuk mempelajari tindakan mana yang tidak diizinkan oleh blok ini, lihat Tind akan, Sumber Daya, dan Kunci Kondisi untuk AWS Secrets Manager. Halaman tersebut menunjukkan bahwa tindakan yang dilakukan pada tipe sumber daya Rahasia mendukung `secretsmanager:ResourceTag/tag-key` kunci kondisi. Beberapa tindakan Secrets Manager tidak mendukung tipe sumber daya tersebut, termasuk `GetRandomPassword` dan `ListSecrets`. Anda harus membuat pernyataan tambahan untuk mengizinkan tindakan tersebut.
- Blok kondisi kedua menjadi benar jika setiap kunci tanda yang diteruskan di permintaan disertakan dalam daftar yang ditentukan. Hal ini dilakukan menggunakan `ForAllValues` dengan operator kondisi `StringEquals`. Jika tidak ada kunci atau subset dari set kunci yang diteruskan, kondisi menjadi benar. Hal ini memungkinkan operasi `Get*` yang tidak memungkinkan tanda diteruskan di permintaan. Jika pemohon menyertakan kunci tanda yang tidak ada dalam daftar, kondisi menjadi salah. Setiap kunci tanda yang diteruskan di permintaan harus sesuai dengan anggota daftar ini. Untuk informasi selengkapnya, lihat Mengatur operator untuk kunci konteks multinilai.
- Blok kondisi ketiga menjadi benar jika permintaan mendukung tanda yang lolos, jika ketiga tanda hadir, dan jika sesuai dengan nilai tanda prinsipal. Blok ini juga menjadi benar jika permintaan tidak mendukung tanda yang diteruskan. Ini berkat `...IfExists` di operator kondisi. Blok menjadi salah jika tidak ada tanda yang diteruskan selama tindakan yang mendukungnya, atau jika kunci dan nilai tanda tidak cocok.

- Pernyataan `AllResourcesSecretsManagerNoTags` memungkinkan tindakan `GetRandomPassword` dan `ListSecrets` yang tidak diperbolehkan oleh pernyataan pertama.
- Pernyataan `ReadSecretsManagerSameTeam` memungkinkan operasi hanya baca jika prinsipal ditandai dengan tanda access-team yang sama sebagai sumber daya. Hal ini diperbolehkan terlepas dari tanda proyek atau tanda pusat biaya.
- Pernyataan `DenyUntagSecretsManagerReservedTags` menolak permintaan untuk menghapus tanda dengan kunci yang dimulai dengan "access-" dari Secrets Manager. Tanda ini digunakan untuk mengontrol akses ke sumber daya, sehingga menghapus tanda dapat menghapus izin.

- Pernyataan `DenyPermissionsManagement` menolak akses untuk membuat, mengedit, atau menghapus kebijakan berbasis sumber daya Secrets Manager. Kebijakan ini dapat digunakan untuk mengubah izin rahasia.

> **Important**
>
> Kebijakan ini menggunakan strategi untuk memungkinkan semua tindakan untuk layanan, tetapi dengan tegas menolak tindakan yang melanggar izin. Menolak tindakan akan membatalkan kebijakan lain yang memungkinkan penanggung jawab melakukan tindakan tersebut. Ini dapat menimbulkan hasil yang tidak diinginkan. Sebagai praktik terbaik, gunakanlah penolakan eksplisit hanya jika tidak ada keadaan yang mengizinkan tindakan tersebut. Jika tidak, izinkan daftar tindakan individu, dan tindakan yang tidak diinginkan ditolak secara default.

Langkah 3: Buat peran

Buat peran IAM berikut dan lampirkan kebijakan **`access-same-project-team`** yang Anda buat pada langkah sebelumnya. Untuk informasi selengkapnya tentang cara membuat peran IAM, lihat Buat peran untuk memberikan izin kepada pengguna IAM. Jika Anda memilih untuk menggunakan federasi daripada pengguna dan peran IAM, lihat Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC.

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
Proyek Teknik Pegasus	rekayasa akses-peg	akses-proyek = `peg` akses tim = `eng` pusat biaya = `987654`	Memungkinkan insinyur membaca semua sumber daya teknik dan membuat serta mengelola sumber daya teknik Pegasus.
Jaminan Kualitas Proyek Pegasus	akses-peg-jaminan kualitas	akses-proyek = `peg`	Memungkinkan tim QA untuk

Fungsi pekerjaan	Nama peran	Tanda peran	Deskripsi peran
		akses tim = `qas` pusat biaya = `987654`	membaca semua sumber daya QA dan membuat, serta mengelola semua sumber daya QA Pegasus.
Proyek Teknik Unicorn	akses-uni-rekayasa	akses-proyek= `uni` akses tim = `eng` pusat biaya = `123456`	Memungkinkan insinyur membaca semua sumber daya teknik dan membuat, serta mengelola sumber daya teknik Unicorn.
Jaminan Kualitas Proyek Unicorn	akses-uni-jaminan kualitas	akses-proyek = `uni` akses tim = `qas` pusat biaya = `123456`	Memungkinkan tim QA untuk membaca semua sumber daya QA dan membuat, serta mengelola semua sumber daya QA Unicorn.

Langkah 4: Uji pembuatan rahasia

Kebijakan izin yang terlampir pada peran memungkinkan karyawan untuk membuat rahasia. Hal ini hanya diperbolehkan jika rahasia ditandai dengan proyek, tim, dan pusat biaya mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan masuk sebagai pengguna Anda, dengan asumsi peran yang benar, dan menguji aktivitas di Secrets Manager.

Untuk mencoba membuat rahasia dengan dan tanpa tag yang diperlukan

1. Di jendela browser utama Anda, tetap masuk sebagai pengguna administrator sehingga Anda dapat meninjau pengguna, peran, dan kebijakan di IAM. Gunakan jendela penyamaran peramban atau peramban terpisah untuk pengujian Anda. Di sana, masuk sebagai pengguna `access-Arnav-peg-eng` IAM dan buka konsol Secrets Manager di https://console.aws.amazon.com/secretsmanager/.
2. Upayakan untuk beralih ke peran `access-uni-engineering`.

   Operasi ini gagal karena nilai tanda `access-project` dan `cost-center` tidak cocok dengan pengguna `access-Arnav-peg-eng` dan peran `access-uni-engineering`.

   Untuk informasi selengkapnya tentang mengganti peran di Konsol Manajemen AWS, lihat Beralih dari pengguna ke peran IAM (konsol)
3. Beralih ke peran `access-peg-engineering`.
4. Simpan rahasia baru menggunakan informasi berikut. Untuk mempelajari cara menyimpan rahasia, lihat Membuat Rahasia Dasar di Panduan Pengguna AWS Secrets Manager .

   > **Important**
   >
   > Manajer Rahasia menampilkan peringatan bahwa Anda tidak memiliki izin untuk AWS layanan tambahan yang berfungsi dengan Manajer Rahasia. Misalnya, untuk membuat kredensial basis data Amazon RDS, Anda harus memiliki izin untuk menjelaskan instans RDS, klaster RDS, dan klaster Amazon Redshift. Anda dapat mengabaikan peringatan ini karena Anda tidak menggunakan AWS layanan khusus ini dalam tutorial ini.

   1. Di bagian Pilih tipe rahasia, pilih Tipe rahasia lainnya. Di dua kotak teks, masukkan `test-access-key` dan `test-access-secret`.
   2. Masukkan `test-access-peg-eng` untuk bidang Nama rahasia.
   3. Tambahkan kombinasi tanda yang berbeda dari tabel berikut dan lihat perilaku yang diharapkan.
   4. Pilih Toko untuk mencoba membuat rahasia. Jika penyimpanan gagal, kembali ke laman konsol Secrets Manager sebelumnya dan gunakan set tag berikutnya dari tabel berikut. Set tag terakhir dizinkan dan akan berhasil membuat rahasia.

Tabel berikut menunjukkan kombinasi tag ABAC untuk `test-access-peg-eng` peran tersebut.

Nilai Tanda **access-project**	Nilai Tanda **access-team**	Nilai Tanda **cost-center**	Tanda tambahan	Perilaku yang diharapkan
(tidak ada)	(tidak ada)	(tidak ada)	(tidak ada)	Ditolak karena nilai tag `access-project` tidak cocok dengan nilai peran `peg`.
`uni`	`eng`	`987654`	(tidak ada)	Ditolak karena nilai tag `access-project` tidak cocok dengan nilai peran `peg`.
`peg`	`qas`	`987654`	(tidak ada)	Ditolak karena nilai tag `access-team` tidak cocok dengan nilai peran `eng`.
`peg`	`eng`	`123456`	(tidak ada)	Ditolak karena nilai tag `cost-center` tidak cocok dengan nilai peran `987654`.
`peg`	`eng`	`987654`	Pemilik = Jane	Ditolak karena tag tambahan `owner` tidak diizinkan oleh kebijakan, meskipun ketiga tanda yang diperlukan hadir dan nilainya sesuai dengan nilai-nilai peran.
`peg`	`eng`	`987654`	Nama = Jane	Diizinkan karena ketiga tanda yang diperlukan ada dan nilainya sesuai dengan nilai-nilai peran. Anda juga diperbolehkan untuk menyertakan tanda `Name`.

5. Keluar dan ulangi tiga langkah pertama dalam prosedur ini untuk setiap peran dan nilai tanda berikut. Pada langkah keempat dalam prosedur ini, uji setiap set tanda yang hilang, tanda

opsional, tanda yang tidak diizinkan, dan nilai tanda tidak valid yang Anda pilih. Kemudian gunakan tanda yang diperlukan untuk membuat rahasia dengan tanda dan nama berikut.

Nama pengguna	Nama peran	Nama rahasia	Tanda rahasia
akses- Mary-peg-qas	akses-peg-jaminan kualitas	uji-akses-peg-qa	akses-proyek = `peg` akses tim = `qas` pusat biaya = `987654`
akses- Saanvi-uni-eng	akses-uni-rekayasa	tes-akses-uni-eng	akses-proyek = `uni` akses tim = `eng` pusat biaya = `123456`
akses- Carlos-uni-qas	akses-uni-jaminan kualitas	test-akses-uni-qas	akses-proyek = `uni` akses tim = `qas` pusat biaya = `123456`

Langkah 5: Uji menampilkan rahasia

Kebijakan yang Anda lampirkan ke setiap peran memungkinkan karyawan melihat rahasia apa pun yang ditandai dengan nama tim mereka, apa pun proyek mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan menguji peran Anda di Secrets Manager.

Untuk uji melihat rahasia dengan dan tanpa tag yang diperlukan

1. Masuk sebagai salah satu pengguna IAM berikut:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`

- `access-Carlos-uni-qas`

2. Beralih ke peran yang sesuai:

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

   Untuk informasi selengkapnya tentang mengganti peran di Konsol Manajemen AWS, lihatBeralih dari pengguna ke peran IAM (konsol).

3. Di panel navigasi sebelah kiri, pilih ikon menu untuk memperluas menu lalu pilih Rahasia.
4. Anda akan melihat keempat rahasia di tabel, apa pun peran Anda saat ini. Ini diharapkan karena kebijakan bernama `access-same-project-team` memungkinkan tindakan `secretsmanager:ListSecrets` untuk semua sumber daya.
5. Pilih nama salah satu rahasia.
6. Pada halaman detail rahasia, tanda peran Anda menentukan apakah Anda dapat melihat isi halaman tersebut. Bandingkan nama peran Anda dengan nama rahasia Anda. Jika mereka memiliki nama tim yang sama, tanda `access-team` cocok. Jika tidak cocok, akses akan ditolak.

   Tabel berikut menunjukkan perilaku tampilan rahasia ABAC untuk setiap peran.

Nama peran	Nama rahasia	Perilaku yang diharapkan
`access-peg-engineering`	`test-access-peg-eng`	Diizinkan
	uji-akses-peg-qa	Ditolak
	tes-akses-uni-eng	Diizinkan
	test-akses-uni-qas	Ditolak
akses-peg-jaminan kualitas	test-akses-peg-eng	Ditolak
	uji-akses-peg-qa	Diizinkan
	tes-akses-uni-eng	Ditolak

Nama peran	Nama rahasia	Perilaku yang diharapkan
	test-akses-uni-qas	Diizinkan
akses-uni-rekayasa	test-akses-peg-eng	Diizinkan
	uji-akses-peg-qa	Ditolak
	tes-akses-uni-eng	Diizinkan
	test-akses-uni-qas	Ditolak
akses-uni-jaminan kualitas	test-akses-peg-eng	Ditolak
	uji-akses-peg-qa	Diizinkan
	tes-akses-uni-eng	Ditolak
	test-akses-uni-qas	Diizinkan

7. Dari breadcrumbs di bagian atas halaman, pilih Rahasia untuk kembali ke daftar rahasia. Ulangi langkah-langkah dalam prosedur ini menggunakan peran yang berbeda untuk menguji apakah Anda dapat melihat masing-masing rahasia.

Langkah 6: Uji skalabilitas

Alasan penting untuk menggunakan kontrol akses berbasis atribut (ABAC) daripada kontrol akses berbasis peran (RBAC) adalah skalabilitas. Saat perusahaan Anda menambahkan proyek, tim, atau orang baru AWS, Anda tidak perlu memperbarui ABAC-driven kebijakan Anda. Misalnya, anggaplah bahwa Example Corporation mendanai proyek baru dengan nama kode Centaur. Seorang insinyur bernama Saanvi Sarkar akan menjadi insinyur utama untuk Centaur sambil terus mengerjakan proyek Unicorn. Saanvi juga akan meninjau pekerjaan untuk proyek Peg. Ada juga beberapa insinyur yang baru direkrut, termasuk Nikhil Jayashankar, yang hanya akan mengerjakan proyek Centaur.

Untuk menambahkan proyek baru ke AWS

1. Masuk sebagai pengguna administrator IAM dan buka konsol IAM di. https://console.aws.amazon.com/iam/

2. Di panel navigasi di sebelah kiri, pilih Per an dan tambahkan peran IAM bernama`access-cen-engineering`. Lampirkan kebijakan **`access-same-project-team`** izin ke peran dan tambahkan tag peran berikut:

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

3. Pada panel navigasi di sebelah kiri, pilih Pengguna.
4. Tambahkan pengguna baru bernama`access-Nikhil-cen-eng`, lampirkan kebijakan bernama`access-assume-role`, dan tambahkan tag pengguna berikut.

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

5. Gunakan prosedur di Langkah 4: Uji pembuatan rahasia dan Langkah 5: Uji menampilkan rahasia. Di jendela peramban lain, uji apakah Nikhil hanya dapat membuat rahasia teknik Centaur, dan apakah dia dapat melihat semua rahasia teknik.
6. Di jendela browser utama tempat Anda masuk sebagai administrator, pilih penggguna`access-Saanvi-uni-eng`.
7. Di tab Izin, hapus kebijakan izin akses-asumsi-peran.
8. Tambahkan kebijakan inline berikut bernama `access-assume-specific-roles`. Untuk informasi selengkapnya tentang menambahkan kebijakan inline ke pengguna, lihat Untuk menyematkan kebijakan inline bagi pengguna atau peran (konsole).

   Kebijakan ABAC: Asumsikan hanya peran tertentu

   Kebijakan ini memungkinkan Saanvi untuk mengambil peran teknik untuk proyek Pegasus dan Centaur. Perlu untuk membuat kebijakan kustom ini karena IAM tidak mendukung tag multinilai. Anda tidak dapat menandai pengguna Saanvi dengan `access-project` = `peg` dan `access-project` = `cen`. Selain itu, model AWS otorisasi tidak dapat mencocokkan kedua nilai tersebut. Untuk informasi selengkapnya, lihat Aturan untuk menandai di IAM dan AWS STS. Sebagai gantinya, Anda harus menentukan secara manual dua peran yang dapat dia tanggung.

   Untuk menggunakan kebijakan ini, ganti teks plac eholder miring dengan informasi akun Anda.

JSON

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

9. Gunakan prosedur di Langkah 4: Uji pembuatan rahasia dan Langkah 5: Uji menampilkan rahasia. Pada jendela peramban lain, konfirmasikan bahwa Saanvi dapat mengambil kedua peran tersebut. Periksa bahwa dia dapat membuat rahasia hanya untuk proyek, tim, dan pusat biaya miliknya, bergantung pada tanda peran. Konfirmasi juga bahwa dia dapat melihat perincian tentang rahasia apa pun yang dimiliki oleh tim teknik, termasuk rahasia yang baru saja dia buat.

Langkah 7: Uji memperbarui dan menghapus rahasia

Kebijakan `access-same-project-team` yang terlampir pada peran memungkinkan karyawan memperbarui dan menghapus rahasia yang ditandai dengan proyek, tim, dan pusat biaya mereka. Konfirmasikan bahwa izin Anda bekerja sesuai harapan dengan menguji peran Anda di Secrets Manager.

Untuk menguji pembaruan dan penghapusan rahasia dengan dan tanpa tag yang diperlukan

1. Masuk sebagai salah satu pengguna IAM berikut:

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

- `access-Nikhil-cen-eng`

2. Beralih ke peran yang sesuai:

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

   Untuk informasi selengkapnya tentang mengganti peran di Konsol Manajemen AWS, lihatBeralih dari pengguna ke peran IAM (konsol).

3. Untuk setiap peran, coba perbarui deskripsi rahasia, lalu coba hapus rahasia berikut. Untuk informasi selengkapnya, lihat, lihat Memodifikasi Rahasia dan Menghapus dan Memulihkan Rahasia di Panduan Pengguna AWS Secrets Manager .

   Tabel berikut menunjukkan perilaku pembaruan dan penghapusan rahasia ABAC untuk setiap peran.

Nama peran	Nama rahasia	Perilaku yang diharapkan
rekayasa akses-peg	test-akses-peg-eng	Diizinkan
	tes-akses-uni-eng	Ditolak
	test-akses-uni-qas	Ditolak
akses-peg-jaminan kualitas	uji-akses-peg-qa	Diizinkan
	tes-akses-uni-eng	Ditolak
akses-uni-rekayasa	tes-akses-uni-eng	Diizinkan
	test-akses-uni-qas	Ditolak
akses-peg-jaminan kualitas	test-akses-uni-qas	Diizinkan

Ringkasan

Anda telah berhasil menyelesaikan semua langkah yang diperlukan untuk menggunakan tanda untuk kontrol akses berbasis atribut (ABAC). Anda telah mempelajari cara mendefinisikan strategi penandaan. Anda menerapkan strategi tersebut ke penanggung jawab dan sumber daya Anda. Anda membuat dan menerapkan kebijakan yang menegakkan strategi untuk Secrets Manager. Anda juga mempelajari bahwa ABAC menskalakan dengan mudah ketika Anda menambahkan proyek dan anggota tim baru. Hasilnya, Anda dapat masuk ke konsol IAM dengan peran pengujian dan pengalaman menggunakan tag untuk masuk ABAC. AWS

> **Note**
>
> Anda menambahkan kebijakan yang memungkinkan tindakan hanya berdasarkan kondisi tertentu. Jika Anda menerapkan kebijakan yang berbeda kepada pengguna atau peran Anda yang memiliki izin lebih luas, maka tindakan tersebut mungkin tidak akan dibatasi untuk memerlukan penandaan. Misalnya, jika Anda memberi pengguna izin administratif penuh menggunakan kebijakan `AdministratorAccess` AWS terkelola, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.

Sumber daya terkait

Untuk informasi terkait, lihat sumber daya berikut:

- Tentukan izin berdasarkan atribut dengan otorisasi ABAC
- AWS kunci konteks kondisi global
- Buat peran untuk memberikan izin kepada pengguna IAM
- Tag untuk AWS Identity and Access Management sumber daya
- Mengontrol akses ke AWS sumber daya menggunakan tag
- Beralih dari pengguna ke peran IAM (konsol)
- Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC

Untuk mempelajari cara memantau tag di akun Anda, lihat Meman tau perubahan tag pada AWS sumber daya dengan alur kerja tanpa server dan Acara Amazon CloudWatch .

Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC

Attribute-based kontrol akses (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Di AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke sumber daya IAM, termasuk entitas IAM (pengguna atau peran), dan ke AWS sumber daya. Ketika entitas digunakan untuk membuat permintaan AWS, mereka menjadi prinsipal dan prinsipal tersebut menyertakan tag.

Anda juga dapat meneruskan tanda sesi saat Anda mengasumsikan peran atau menggabungkan pengguna. Anda dapat menentukan kebijakan yang menggunakan kunci tanda kondisi untuk memberikan izin kepada prinsipal Anda berdasarkan tanda mereka. Saat Anda menggunakan tanda untuk mengontrol akses ke sumber daya AWS , Anda memungkinkan tim dan sumber daya Anda tumbuh dengan lebih sedikit perubahan ke kebijakan AWS . Kebijakan ABAC lebih fleksibel daripada AWS kebijakan tradisional, yang mengharuskan Anda mencantumkan setiap sumber daya individu. Untuk informasi selengkapnya tentang ABAC dan manfaatnya dibandingkan kebijakan tradisional, lihat Tentukan izin berdasarkan atribut dengan otorisasi ABAC.

Jika perusahaan Anda menggunakan penyedia SAML-based identitas (IdP) untuk mengelola identitas pengguna perusahaan, Anda dapat menggunakan atribut SAML untuk kontrol akses yang halus di. AWS Atribut dapat mencakup pengidentifikasi pusat biaya, alamat surel pengguna, klasifikasi departemen, dan penugasan proyek. Saat Anda meneruskan atribut tersebut sebagai tanda sesi, Anda dapat mengontrol akses ke AWS berdasarkan tanda sesi ini.

Untuk menyelesaikan Tutorial ABAC dengan meneruskan atribut SAML ke penanggung jawab sesi Anda, selesaikan tugas di Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag, dengan perubahan yang disertakan di topik ini.

Prasyarat

Untuk melakukan langkah-langkah menggunakan tanda sesi SAML untuk ABAC, Anda harus sudah memiliki hal-hal berikut:

- Akses ke SAML-based IdP tempat Anda dapat membuat pengguna uji dengan atribut tertentu.
- Kemampuan untuk masuk sebagai pengguna dengan izin administratif.
- Pengalaman membuat dan mengedit pengguna, peran, dan kebijakan IAM di Konsol Manajemen AWS. Tetapi, jika Anda memerlukan bantuan untuk mengingat proses manajemen IAM, tutorial ABAC memberikan tautan yang bisa Anda gunakan untuk melihat instruksi langkah demi langkah.
- Pengalaman menyiapkan SAML-based IdP di IAM. Untuk melihat perincian lebih lanjut dan tautan ke dokumentasi IAM terperinci, lihat Melewati tag sesi menggunakan AssumeRoleWithSAML.

Langkah 1: Buat pengguna uji

Lewati petunjuk di Langkah 1: Buat pengguna uji. Karena identitas Anda ditentukan di penyedia, Anda tidak perlu menambahkan pengguna IAM untuk karyawan Anda.

Langkah 2: Buat kebijakan ABAC

Ikuti petunjuk di Langkah 2: Buat kebijakan ABAC untuk membuat kebijakan pengelolaan khusus di IAM.

Langkah 3: Buat dan konfigurasi peran SAML

Saat Anda menggunakan tutorial ABAC untuk SAML, Anda harus melakukan langkah-langkah tambahan untuk membuat peran, mengkonfigurasi SAML IdP, dan mengaktifkan akses. Konsol Manajemen AWS Untuk informasi selengkapnya, lihat Langkah 3: Buat peran.

Langkah 3A: Buat peran SAML.

Buat peran tunggal yang memercayai penyedia identitas SAML Anda dan pengguna `test-session-tags` yang Anda buat pada langkah 1. Tutorial ABAC menggunakan peran yang berbeda dengan tanda peran yang berbeda. Karena Anda meneruskan tanda sesi dari IdP SAML Anda, Anda hanya perlu satu peran. Untuk mempelajari cara membuat SAML-based peran, lihatBuat peran untuk federasi SAML 2.0 (konsol).

Beri nama peran `access-session-tags`. Lampirkan kebijakan izin `access-same-project-team` ke peran tersebut. Edit kebijakan kepercayaan peran untuk menggunakan kebijakan berikut. Untuk petunjuk terperinci tentang cara mengedit hubungan kepercayaan dari suatu peran, lihat Memperbarui kebijakan kepercayaan peran.

Kebijakan kepercayaan peran berikut memungkinkan penyedia identitas SAML Anda dan pengguna `test-session-tags` untuk mengambil peran tersebut. Saat mereka mengambil peran tersebut, mereka harus meneruskan tiga tanda sesi khusus. Diperlukan tindakan `sts:TagSession` untuk memungkinkan penerusan tanda sesi.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

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

Per `AllowSamlIdentityAssumeRole` nyataan tersebut memungkinkan anggota tim Teknik dan Jaminan Kualitas untuk mengambil peran ini ketika mereka berfederasi AWS dari Example Corporation IdP. Penye `ExampleCorpProvider` dia SAML didefinisikan dalam IAM. Administrator telah mengatur pernyataan SAML untuk meneruskan tiga tanda sesi yang diperlukan. Pernyataan ini dapat meneruskan tanda tambahan, tetapi ketiga tanda ini harus ada. Atribut identitas dapat memiliki nilai apa pun untuk tanda `cost-center` dan `access-project`. Tetapi, nilai atribut `access-team` harus cocok dengan `eng` atau `qas` untuk menunjukkan bahwa identitas ada pada tim Engineering atau Jaminan Kualitas.

Langkah 3B: Mengonfigurasi IdP SAML

Konfigurasikan IdP SAML Anda untuk meneruskan atribut `cost-center`, `access-project`, dan `access-team` sebagai tanda sesi. Untuk informasi selengkapnya, lihat Melewati tag sesi menggunakan AssumeRoleWithSAML.

Untuk meneruskan atribut ini sebagai tanda sesi, sertakan elemen berikut dalam pernyataan SAML Anda.

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

Langkah 3C: Aktifkan akses konsol

Aktifkan akses konsol untuk pengguna SAML gabungan Anda. Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS.

Langkah 4: Uji pembuatan rahasia

Bergabunglah ke dalam Konsol Manajemen AWS menggunakan `access-session-tags` peran. Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS. Kemudian ikuti petunjuk di Langkah 4: Uji pembuatan rahasia untuk membuat rahasia. Gunakan identitas SAML yang berbeda dengan atribut untuk mencocokkan tanda yang ditunjukkan dalam tutorial ABAC. Untuk informasi selengkapnya, lihat Langkah 4: Uji pembuatan rahasia.

Langkah 5: Uji menampilkan rahasia

Ikuti petunjuk di Langkah 5: Uji menampilkan rahasia untuk melihat rahasia yang Anda buat di langkah sebelumnya. Gunakan identitas SAML yang berbeda dengan atribut untuk mencocokkan tanda yang ditunjukkan dalam tutorial ABAC.

Langkah 6: Uji skalabilitas

Ikuti petunjuk di Langkah 6: Uji skalabilitas untuk menguji skalabilitas. Lakukan ini dengan menambahkan identitas baru di SAML-based IdP Anda dengan atribut berikut:

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

Langkah 7: Uji memperbarui dan menghapus rahasia

Ikuti petunjuk di Langkah 7: Uji memperbarui dan menghapus rahasia untuk memperbarui dan menghapus rahasia. Gunakan identitas SAML yang berbeda dengan atribut untuk mencocokkan tanda yang ditunjukkan dalam tutorial ABAC.

> **Important**
>
> Hapus semua rahasia yang Anda buat untuk menghindari biaya tagihan. Untuk detail tentang penetapan harga di Secrets Manager, lihat AWS Secrets Manager Harga.

Ringkasan

Anda telah berhasil menyelesaikan semua langkah yang diperlukan untuk menggunakan tanda sesi SAML dan tanda sumber daya untuk manajemen izin.

> **Note**
>
> Anda menambahkan kebijakan yang memungkinkan tindakan hanya berdasarkan kondisi tertentu. Jika Anda menerapkan kebijakan yang berbeda kepada pengguna atau peran Anda yang memiliki izin lebih luas, maka tindakan tersebut mungkin tidak akan dibatasi untuk memerlukan penandaan. Misalnya, jika Anda memberi pengguna izin administratif penuh menggunakan kebijakan `AdministratorAccess` AWS terkelola, kebijakan ini tidak membatasi akses tersebut. Untuk informasi selengkapnya tentang bagaimana izin ditentukan saat beberapa kebijakan dilibatkan, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.

Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka

Anda dapat mengizinkan pengguna untuk mengelola perangkat dan kredensialnya sendiri multi-faktor autentikasi (MFA) di halaman Kredensial Keamanan. Anda dapat menggunakan Konsol Manajemen AWS untuk mengonfigurasi kredensi (kunci akses, kata sandi, sertifikat penandatanganan, dan kunci publik SSH), menghapus atau menonaktifkan kredenal yang tidak diperlukan, dan mengaktifkan perangkat MFA untuk pengguna Anda. Ini berguna untuk sejumlah kecil pengguna, tetapi tugas

itu dapat dengan cepat memakan waktu seiring bertambahnya jumlah pengguna. Tutorial ini menunjukkan cara mengaktifkan praktik terbaik tersebut tanpa membebani administrator Anda.

Tutorial ini menunjukkan cara mengizinkan pengguna mengakses AWS layanan, tetapi hanya ketika mereka masuk dengan MFA. Jika mereka belum masuk dengan perangkat MFA, maka pengguna tidak dapat mengakses layanan lainnya.

Alur kerja ini memiliki tiga langkah dasar.

Langkah 1: Buat kebijakan untuk menerapkan masuk MFA

Buat kebijakan terkelola pelanggan yang melarang semua tindakan kecuali beberapa tindakan IAM. Pengecualian ini memungkinkan pengguna untuk mengubah kredensialnya sendiri dan mengelola perangkat MFA mereka di halaman kredensi Keamanan. Untuk informasi selengkapnya tentang mengakses halaman tersebut, lihat Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol).

Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda

Buat grup pengguna yang anggotanya memiliki akses penuh ke semua tindakan Amazon EC2 jika mereka masuk dengan MFA. Untuk membuat grup pengguna seperti itu, Anda melampirkan kebijakan AWS terkelola yang disebut `AmazonEC2FullAccess` dan kebijakan terkelola pelanggan yang Anda buat pada langkah pertama.

Langkah 3: Uji akses pengguna Anda

Masuk sebagai pengguna uji untuk memverifikasi bahwa akses ke Amazon EC2 diblokir hingga pengguna membuat perangkat MFA. Pengguna kemudian dapat masuk menggunakan perangkat tersebut.

Prasyarat

Untuk melakukan langkah-langkah di tutorial ini, Anda harus sudah memiliki hal-hal berikut:

- Sebuah Akun AWS yang dapat Anda masuki sebagai pengguna IAM dengan izin administratif.
- Nomor ID akun Anda, yang Anda ketikkan ke dalam kebijakan di Langkah 1.

  Untuk menemukan nomor ID akun Anda, di bilah navigasi di bagian atas halaman, pilih Dukungan lalu pilih Pusat Dukungan. Anda dapat menemukan ID akun Anda di menu Dukungan di halaman ini.

- Perangkat MFA virtual (berbasis perangkat lunak), kunci keamanan FIDO, atau perangkat MFA berbasis perangkat keras.
- Uji pengguna IAM yang merupakan anggota grup pengguna sebagai berikut:

Nama pengguna	Instruksi nama pengguna	Nama grup pengguna	Tambahkan pengguna sebagai anggota	Instruksi grup pengguna
MFAUser	Pilih hanya opsi untuk Aktifkan akses konsol — opsional, dan tetapkan kata sandi.	EC2MFA	MFAUser	JANGAN lampirkan kebijakan apa pun atau memberikan izin ke grup pengguna ini.

Langkah 1: Buat kebijakan untuk menerapkan masuk MFA

Anda memulai dengan membuat kebijakan terkelola pelanggan IAM yang menolak semua izin kecuali izin yang diperlukan oleh pengguna IAM untuk mengelola kredensial dan perangkat MFA mereka sendiri.

1. Masuk ke Konsol AWS Manajemen sebagai pengguna dengan kredensi administrator. Untuk mematuhi praktik terbaik IAM, jangan masuk dengan Pengguna root akun AWS kredensi Anda.

   > **Important**
   >
   > Praktik terbaik IAM merekomendasikan agar Anda mengharuskan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara alih-alih menggunakan pengguna IAM dengan kredensi jangka panjang. Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

2. Buka konsol IAM di https://console.aws.amazon.com/iam/.

3. Pada panel navigasi, pilih Kebijakan, lalu pilih Buat kebijakan.

4. Pilih tab JSON dan salin teks dari dokumen kebijakan JSON berikut: AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.
5. Tempelkan teks kebijakan ke dalam kotak teks JSON. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

> **Note**
>
> Anda dapat beralih antara editor Visual dan opsi JSON kapan saja. Namun, kebijakan di atas mencakup unsur `NotAction` yang tidak didukung di editor visual. Untuk kebijakan ini, Anda akan melihat pemberitahuan di tab Editor visual. Kembali ke JSON untuk terus bekerja dengan kebijakan ini.
> Kebijakan contoh ini tidak mengizinkan pengguna untuk mengatur ulang kata sandi saat masuk Konsol Manajemen AWS untuk pertama kalinya. Kami menyarankan Anda untuk tidak memberikan izin kepada pengguna baru sampai setelah mereka masuk dan mengatur ulang kata sandi mereka.

6. Pada halaman Tinjau dan buat, ketik **`Force_MFA`** nama kebijakan. Untuk deskripsi kebijakan, ketik **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** Di area Tag, Anda dapat menambahkan pasangan nilai kunci tag secara opsional ke kebijakan yang dikelola pelanggan. Tinjau izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul di daftar kebijakan terkelola dan siap dilampirkan.

Langkah 2: Lampirkan kebijakan ke grup pengguna uji Anda

Selanjutnya Anda melampirkan dua kebijakan ke grup pengguna IAM pengujian, yang akan digunakan untuk memberikan izin yang dilindungi MFA.

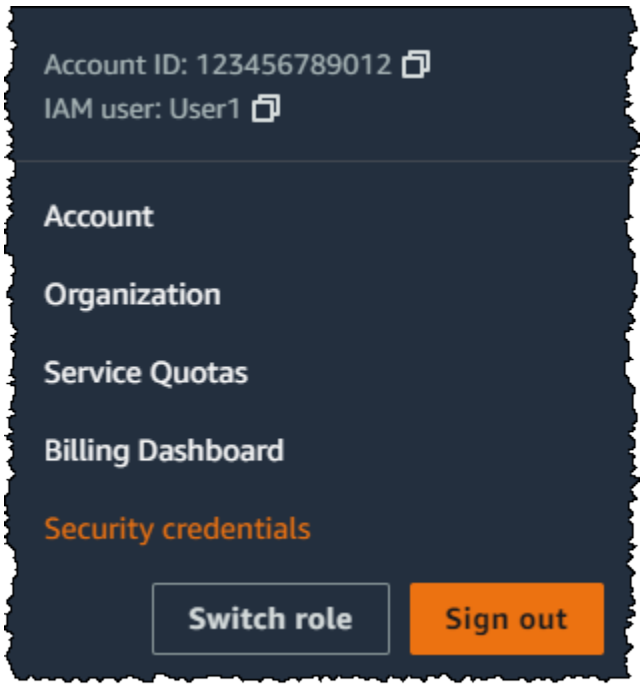
1. Pada panel navigasi, pilih User groups (Grup pengguna).
2. Di kotak pencarian, ketik **`EC2MFA`**, lalu pilih nama grup (bukan kotak centang) dalam daftar.
3. Pilih tab Izin, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.

4. Pada halaman Lampirkan kebijakan izin ke grup EC2 MFA, di kotak pencarian, ketik. **`EC2Full`** Kemudian pilih kotak centang di sebelah Amazon EC2 FullAccess dalam daftar. Jangan menyimpan perubahan Anda.
5. Di kotak pencarian, ketik**`Force`**, lalu pilih kotak centang di sebelah Force_MFA dalam daftar.
6. Pilih Lampirkan kebijakan.

Langkah 3: Uji akses pengguna Anda

Di bagian tutorial ini, Anda masuk sebagai pengguna uji dan memverifikasi bahwa kebijakan berjalan sebagaimana mestinya.

1. Masuk ke Anda Akun AWS sebagai **`MFAUser`** dengan kata sandi yang Anda tetapkan di bagian sebelumnya. Gunakan URL: `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Pilih EC2 untuk membuka konsol Amazon EC2 dan memverifikasi bahwa pengguna tidak memiliki izin untuk melakukan apa pun.
3. Di bilah navigasi di kanan atas, pilih nama `MFAUser` pengguna, lalu pilih Kredensi keamanan.



4. Sekarang, tambahkan perangkat MFA. Di bagian Multi-Factor Authentication (MFA), pilih Tetapkan perangkat MFA.

> **Note**
>
> Anda mungkin menerima kesalahan bahwa Anda tidak berwenang untuk melakukan `iam:DeleteVirtualMFADevice`. Hal ini dapat terjadi jika sebelumnya seseorang mulai menetapkan perangkat MFA virtual ke pengguna ini dan membatalkan proses. Untuk melanjutkan, Anda atau administrator lain harus menghapus perangkat MFA virtual pengguna yang belum ditetapkan. Untuk informasi selengkapnya, lihat Saya tidak berwenang untuk melakukan: iam: DeleteVirtualMFADevice.

5. Untuk tutorial ini, kami menggunakan perangkat MFA virtual (berbasis perangkat lunak), seperti aplikasi Google Authenticator di ponsel. Pilih aplikasi Authenticator, lalu klik Berikutnya.

   IAM menghasilkan dan menampilkan informasi konfigurasi untuk perangkat MFA virtual, termasuk grafik kode QR. Grafik adalah representasi kunci konfigurasi rahasia yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.

6. Buka aplikasi MFA virtual Anda. (Untuk daftar aplikasi yang dapat Anda gunakan untuk meng-host perangkat MFA virtual, lihat Aplikasi MFA Virtual.) Jika aplikasi MFA virtual mendukung beberapa akun (beberapa perangkat MFA virtual), pilih opsi untuk membuat akun baru (perangkat MFA virtual baru).
7. Tentukan apakah aplikasi MFA mendukung kode QR, kemudian lakukan salah satu hal berikut:

   - Dari wizard, pilih Tampilkan kode QR. Lalu gunakan aplikasi untuk memindai kode QR. Misalnya, Anda dapat memilih ikon kamera atau memilih opsi yang mirip dengan Pindai kode, lalu gunakan kamera perangkat untuk memindai kode.
   - Di wizard Siapkan perangkat, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam aplikasi MFA Anda.

   Saat Anda selesai, perangkat MFA virtual mulai membuat kata sandi sekali pakai.

8. Di wizard Menyiapkan perangkat, di Masukkan kode dari aplikasi autentikator Anda. kotak, ketik kata sandi satu kali yang saat ini muncul di perangkat MFA virtual. Pilih Daftar MFA.

> **Important**
>
> Kirim permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, perangkat MFA berhasil dikaitkan dengan pengguna. Namun, perangkat MFA tidak sinkron. Hal ini terjadi

karena kata sandi sekali pakai berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat MFA virtual sekarang siap digunakan. AWS

9. Keluar dari konsol lalu masuk sebagai **`MFAUser`** lagi. Kali ini AWS meminta Anda untuk kode MFA dari ponsel Anda. Saat Anda mendapatkannya, ketik kode di kotak, lalu pilih Kirim.
10. Pilih EC2 untuk membuka konsol Amazon EC2 lagi. Perhatikan bahwa saat ini Anda dapat melihat semua informasi dan melakukan tindakan apa pun yang Anda inginkan. Jika Anda mengunjungi konsol lain sebagai pengguna ini, Anda melihat akses pesan yang ditolak. Alasannya adalah kebijakan dalam tutorial ini hanya dapat diakses oleh Amazon EC2

Sumber daya terkait

Untuk informasi tambahan, lihat topik berikut:

- AWS Multi-factor otentikasi di IAM
- MFA mengaktifkan login

Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP)

Untuk mengatur federasi SAML untuk AWS akun Anda, Anda perlu membuat SAML Identity Provider (IdP). Tutorial ini menunjukkan cara menggunakan CloudFormation template untuk membuat SAML IdP yang membangun kepercayaan antara AWS dan IdP eksternal Anda.

Template membuat SAML IdP yang dikonfigurasi dengan dokumen metadata IdP Anda. Peran IAM federasi kemudian dapat mereferensikan IdP ini untuk memungkinkan pengguna yang diautentikasi dari IdP eksternal Anda mengakses sumber daya. AWS

Sumber daya yang digunakan terdiri dari IdP SAML yang dikonfigurasi dengan dokumen metadata IdP Anda dan pengaturan enkripsi opsional.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Python 3.6 atau yang lebih baru diinstal pada mesin lokal Anda untuk menjalankan perintah Python yang digunakan dalam tutorial ini untuk memformat file XML metadata SAML IdP Anda.
- Dokumen metadata SAML dari IdP eksternal Anda disimpan sebagai file XML.

Buat IdP SAML menggunakan CloudFormation

Untuk membuat SAML IdP, buat CloudFormation template dan gunakan untuk membuat tumpukan yang berisi sumber daya IdP.

Buattemplat

Pertama, buat CloudFormation template.

1. Di Templat bagian tersebut, pilih ikon salin pada tab JSON atau YAML untuk menyalin konten template.
2. Tempelkan isi template ke file baru.
3. Simpan file secara lokal.

Buat tumpukan

Selanjutnya, gunakan template yang telah Anda simpan untuk menyediakan tumpu CloudFormation kan.

1. Buka CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada halaman Stacks, dari menu Buat tumpukan, pilih dengan sumber daya baru (standar).
3. Tentukan template:
    a. Di bawah Prasyarat, pilih Pilih template yang ada.
    b. Di bawah Tentukan template, pilih Unggah file template.
    c. Pilih Pilih file, navigasikan ke file template, dan pilih.
    d. Pilih Berikutnya.
4. Tentukan detail tumpukan berikut:
    a. Masukkan nama tumpukan.
    b. Untuk IdentityProviderName, Anda dapat membiarkan ini kosong untuk membuat nama secara otomatis berdasarkan nama tumpukan, atau memasukkan nama khusus untuk IdP

SAML Anda. Nama khusus hanya harus berisi karakter alfanumerik, titik, garis bawah, dan tanda hubung.

c. Untuk IdentityProviderSAMLMetadataDocument, Anda perlu memformat file XML metadata SAML Anda sebagai satu baris sebelum menempelkannya ke bidang ini. Ini diperlukan karena CloudFormation konsol memerlukan konten XML untuk diformat sebagai satu baris ketika melewati parameter konsol.

Gunakan perintah Python berikut untuk memformat ulang file XML Anda:

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
metadata.xml
```

> **Note**
>
> Dokumen metadata SAML IdP harus diformat sebagai satu baris untuk input parameter konsol. Perintah Python menghapus jeda baris dan spasi putih ekstra untuk membuat format yang diperlukan sambil mempertahankan semua konten dan struktur asli.

Salin output dari perintah Python dan tempelkan ke IdentityProviderSAMLMetadataDocument bidang.

Contoh dokumen metadata SAML yang diformat (disingkat):

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
```

```
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Untuk parameter lain, terima nilai default atau masukkan sendiri berdasarkan kebutuhan Anda:

- IdentityProviderAddPrivateKey- Kunci pribadi opsional untuk mendekripsi pernyataan SAML
- IdentityProviderAssertionEncryptionMode- Opsional, mengatur mode enkripsi untuk pernyataan SAML (Diizinkan, Diperlukan, atau kosong)

e. Pilih Berikutnya.

5. Konfigurasikan opsi tumpukan:

a. Di bawah Opsi kegagalan tumpukan, pilih Hapus semua sumber daya yang baru dibuat.

> **Note**
>
> Memilih opsi ini mencegah Anda dari kemungkinan ditagih untuk sumber daya yang kebijakan penghapusannya menentukan bahwa sumber daya tersebut akan dipertahankan meskipun pembuatan tumpukan gagal.

b. Terima semua nilai default lainnya.

c. Di bawah Kemampuan, centang kotak untuk mengetahui bahwa CloudFormation mungkin membuat sumber daya IAM di akun Anda.

d. Pilih Berikutnya.

6. Tinjau detail tumpukan dan pilih Kirim.

CloudFormation membuat tumpukan. Setelah pembuatan tumpukan selesai, sumber daya tumpukan siap digunakan. Anda dapat menggunakan tab Resources pada halaman detail tumpukan untuk melihat sumber daya yang disediakan di akun Anda.

Tumpukan akan menampilkan nilai-nilai berikut, yang dapat Anda lihat di tab Output:

- ProviderArn: ARN dari IdP SAML yang dibuat (misalnya,). `arn:aws:iam::123456789012:saml-provider/CompanyIdP` Anda akan membutuhkan ARN ini saat membuat peran yang mempercayai penyedia ini.

- ProviderName: Nama IdP SAML yang dibuat (misalnya, `CompanyIdP` jika Anda menentukan nama khusus, atau `my-saml-stack-saml-provider` jika Anda menggunakan penamaan default).

Output ini juga diekspor, memungkinkannya diimpor oleh tumpu CloudFormation kan lain menggunakan `Fn::ImportValue` fungsi.

Verifikasi SAML IdP

Setelah SAML IdP dibuat, Anda dapat memverifikasi konfigurasinya dan mencatat ARN-nya untuk digunakan dengan peran federasi.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Penyedia identitas.

   Anda akan melihat IdP SAML yang baru dibuat dalam daftar.
3. Pilih nama IdP untuk melihat detailnya.

   Pada halaman detail IdP, Anda dapat melihat dokumen metadata SAML dan detail konfigurasi lainnya.
4. Perhatikan ARN Penye dia yang ditampilkan pada halaman detail.

   Anda akan memerlukan ARN ini saat membuat peran IAM federasi yang mempercayai IdP ini.
5. Tinjau dokumen metadata untuk memastikannya cocok dengan apa yang Anda berikan dari IdP eksternal Anda.

IdP SAML Anda sekarang siap digunakan oleh peran IAM federasi. Anda dapat membuat peran yang mempercayai IdP ini untuk mengizinkan pengguna yang diautentikasi dari IdP eksternal Anda mengambil peran tersebut dan mengakses sumber daya tersebut. AWS

Bersihkan: hapus sumber daya

Sebagai langkah terakhir, Anda akan menghapus tumpukan dan sumber daya yang dikandungnya.

1. Buka CloudFormation konsol.
2. Pada halaman Stacks, pilih tumpukan yang dibuat dari template, lalu pilih Hapus, lalu kon firmasikan H apus.

   CloudFormation memulai penghapusan tumpukan dan semua sumber daya yang disertakan.

CloudFormation rincian template

Sumber daya

Tem CloudFormation plate untuk tutorial ini akan membuat sumber daya berikut di akun Anda:

`AWS::IAM::SAMLProvider`: IdP SAML yang membangun kepercayaan antara AWS dan IdP eksternal Anda.

Konfigurasi

Template mencakup parameter yang dapat dikonfigurasi berikut:

- IdentityProviderName- Nama untuk IdP SAML Anda (biarkan kosong untuk nama yang dibuat secara otomatis)

  Contoh: `CompanyIdP` atau `EnterpriseSSO`
- IdentityProviderSAMLMetadataDocument- Dokumen metadata SAML dari IdP eksternal Anda (diformat sebagai satu baris)
- IdentityProviderAddPrivateKey- Kunci pribadi opsional untuk mendekripsi pernyataan SAML
- IdentityProviderAssertionEncryptionMode- Opsional, mengatur mode enkripsi untuk pernyataan SAML

CloudFormation templat

Simpan kode JSON atau YAML berikut sebagai file terpisah untuk digunakan sebagai CloudFormation template untuk tutorial ini.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
```

```
      "ConstraintDescription": "Must be empty or contain only alphanumeric
characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    }
  },
  "Conditions": {
    "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
    "HasEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]},
    "HasCustomName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "IdentityProviderName"},
""]}]}
  },
  "Resources": {
    "SAMLProvider": {
      "Type": "AWS::IAM::SAMLProvider",
      "Properties": {
        "Name": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Ref": "AWS::NoValue"}]},
        "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Fn::Sub": "${AWS::StackName}-saml-provider"}]}
          }
        ],
```

```
        "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
 "IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
        "AssertionEncryptionMode": {"Fn::If": ["HasEncryptionMode", {"Ref":
 "IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
      }
    }
  },
  "Outputs": {
    "ProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderARN"}
      }
    },
    "ProviderName": {
      "Description": "Name of the SAML Identity Provider",
      "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
 {"Fn::Sub": "${AWS::StackName}-saml-provider"}]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderName"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens'

  IdentityProviderSAMLMetadataDocument:
    Type: String
```

```
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

Conditions:
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasEncryptionMode: !Not [!Equals [!Ref IdentityProviderAssertionEncryptionMode,
 ""]]
  HasCustomName: !Not [!Equals [!Ref IdentityProviderName, ""]]

Resources:
  SAMLProvider:
    Type: 'AWS::IAM::SAMLProvider'
    Properties:
      Name: !If
        - HasCustomName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      Tags:
        - Key: Name
          Value: !If
            - HasCustomName
            - !Ref IdentityProviderName
            - !Sub '${AWS::StackName}-saml-provider'
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
```

```
        - HasEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

Outputs:
  ProviderARN:
    Description: 'ARN of the created SAML Identity Provider'
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-ProviderARN'

  ProviderName:
    Description: 'Name of the SAML Identity Provider'
    Value: !If
      - HasCustomName
      - !Ref IdentityProviderName
      - !Sub '${AWS::StackName}-saml-provider'
    Export:
      Name: !Sub '${AWS::StackName}-ProviderName'
```

Tutorial IAM: Gunakan CloudFormation template untuk membuat peran IAM federasi SAMP

Jika Anda memiliki Penyedia Identitas SAMP (IDP) yang sudah dikonfigurasi di akun AWS Anda, Anda dapat membuat peran IAM federasi yang mempercayai IDP tersebut. Tutorial ini menunjukkan cara menggunakan CloudFormation template untuk membuat peran IAM federasi SAMP yang dapat diasumsikan oleh pengguna yang diautentikasi melalui IDP eksternal Anda.

Template membuat peran IAM federasi dengan kebijakan kepercayaan yang memungkinkan IDP SAMP Anda untuk mengambil peran tersebut. Pengguna yang diautentikasi oleh iDP eksternal Anda dapat mengambil peran ini untuk AWS mengakses sumber daya berdasarkan izin yang dilampirkan pada peran tersebut.

Sumber daya yang digunakan terdiri dari yang berikut:

- Peran IAM federasi yang mempercayai IDP SAMP Anda yang ada.
- Kebijakan terkelola yang dapat dikonfigurasi yang dapat dilampirkan ke peran untuk memberikan izin tertentu.
- Pengaturan batas izin opsional dan durasi sesi.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- IDP SAMP yang sudah ada dikonfigurasi di AWS akun Anda. Jika Anda tidak memilikinya, Anda dapat membuatnya menggunakan Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP) tutorial.
- ARN dari SAMP iDP Anda, yang harus Anda tentukan sebagai parameter saat membuat tumpukan.
- Python 3.6 atau yang lebih baru diinstal pada mesin lokal Anda untuk menjalankan perintah Python yang digunakan dalam tutorial ini untuk memformat file XHTML metadata SAMP IDP Anda.

Buat peran federasi SAMP menggunakan CloudFormation

Untuk membuat peran federasi SAMP, Anda akan membuat CloudFormation template dan menggunakannya untuk membuat tumpukan yang berisi peran.

Buattemplat

Pertama, buat CloudFormation template.

1. Di Templat bagian ini, klik ikon salin pada tab JSON atau YAMAL untuk menyalin konten templat.
2. Tempelkan konten template ke file baru.
3. Simpan file secara lokal.

Buat tumpukan

Selanjutnya, gunakan template yang telah Anda simpan untuk menyediakan CloudFormation tumpukan.

1. Buka CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada halaman Stacks, dari menu Create stack, pilih dengan sumber daya baru (standar).
3. Tentukan template:
    a. Di bawah Prasyarat, pilih Pilih templat yang ada.
    b. Di bawah Tentukan templat, pilih Unggah file templat.
    c. Pilih Pilih file, navigasikan ke file template, dan pilih.

d. Pilih Berikutnya.

4. Tentukan detail tumpukan berikut:

   a. Masukkan nama tumpukan.

   b. Untuk SamlProviderARN, masukkan ARN dari IDP SAMP Anda yang ada. Ini harus dalam format`arn:aws:iam::123456789012:saml-provider/YourProviderName`.

      Contoh: `arn:aws:iam::123456789012:saml-provider/CompanyIdP`

      > **Note**
      >
      > Jika Anda membuat SAMP IDP Anda menggunakan Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP) tutorial, Anda dapat menemukan penyedia ARN di tab Output dari tumpukan itu. CloudFormation

   c. Untuk RoleName, Anda dapat membiarkan ini kosong untuk menghasilkan nama secara otomatis berdasarkan nama tumpukan, atau memasukkan nama khusus untuk peran IAM.

      Contoh: `SAML-Developer-Access` atau `SAML-ReadOnly-Role`

   d. Untuk parameter lain, terima nilai default atau masukkan sendiri berdasarkan kebutuhan Anda:

      - RoleSessionDuration- Durasi sesi maksimum dalam hitungan detik (3600-43200, default 7200)

        Contoh: `14400` (4 jam)

      - RolePermissionsBoundary- ARN opsional dari kebijakan batas izin

        Contoh: `arn:aws:iam::123456789012:policy/DeveloperBoundary`

      - RolePath- Jalur untuk peran IAM (default adalah/)

        Contoh: `/saml-roles/`

      - ManagedPolicy1-5- ARN opsional hingga 5 kebijakan terkelola untuk dilampirkan

        Contoh untuk ManagedPolicy 1: `arn:aws:iam::aws:policy/ReadOnlyAccess`

        Contoh untuk ManagedPolicy 2: `arn:aws:iam::123456789012:policy/CustomPolicy`

e. Pilih Berikutnya.

5. Konfigurasikan opsi tumpukan:

   a. Di bawah Opsi kegagalan tumpukan, pilih Hapus semua sumber daya yang baru dibuat.

   > **Note**
   >
   > Memilih opsi ini mencegah Anda dari kemungkinan ditagih untuk sumber daya yang kebijakan penghapusan menentukan mereka dipertahankan bahkan jika pembuatan tumpukan gagal.

   b. Terima semua nilai default lainnya.

   c. Di bawah Kemampuan, centang kotak untuk mengetahui bahwa CloudFormation mungkin membuat sumber daya IAM di akun Anda.

   d. Pilih Berikutnya.

6. Tinjau detail tumpukan dan pilih Kirim.

CloudFormation menciptakan tumpukan. Setelah pembuatan tumpukan selesai, sumber daya tumpukan siap digunakan. Anda dapat menggunakan tab Sumber Daya di halaman detail tumpukan untuk melihat sumber daya yang disediakan di akun Anda.

Tumpukan akan menampilkan nilai berikut, yang dapat Anda lihat di tab Output:

- roLearn: ARN dari peran IAM yang dibuat (misalnya, `arn:aws:iam::123456789012:role/SAML-Developer-Access` atau `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` jika menggunakan nama yang dibuat secara otomatis).

Anda akan memerlukan ARN peran ini saat mengonfigurasi IDP Anda untuk mengirim atribut SAMP yang sesuai untuk asumsi peran.

Uji peran federasi SAMP

Setelah peran federasi SAMP dibuat, Anda dapat memverifikasi konfigurasinya dan menguji pengaturan federasi.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.

3. Temukan dan pilih peran federasi Anda yang baru dibuat.

   Jika Anda memberikan nama peran khusus, cari nama itu. Jika Anda membiarkan RoleName parameter kosong, peran akan memiliki nama yang dibuat secara otomatis berdasarkan nama tumpukan dan pengidentifikasi unik.
4. Pilih tab Hubungan kepercayaan untuk meninjau kebijakan kepercayaan.

   Kebijakan kepercayaan harus menunjukkan bahwa IDP SAMP Anda dipercaya untuk mengambil peran ini dengan syarat bahwa audiens `SAML:aud` SAMP () cocok. `https://signin.aws.amazon.com/saml`
5. Pilih tab Izin untuk meninjau kebijakan terlampir.

   Anda dapat melihat kebijakan terkelola apa pun yang dilampirkan pada peran selama pembuatan.
6. Perhatikan ARN Peran yang ditampilkan di halaman ringkasan peran.

   Anda akan memerlukan ARN ini untuk mengonfigurasi iDP eksternal Anda agar pengguna dapat mengambil peran ini.

Peran federasi SAMP Anda sekarang siap digunakan. Konfigurasikan IDP eksternal Anda untuk menyertakan ARN peran ini dalam pernyataan SAMP, dan pengguna yang diautentikasi akan dapat mengambil peran ini untuk mengakses sumber daya. AWS

Bersihkan: hapus sumber daya

Sebagai langkah terakhir, Anda akan menghapus tumpukan dan sumber daya yang dikandungnya.

1. Buka CloudFormation konsol.
2. Pada halaman Stacks, pilih tumpukan yang dibuat dari templat, lalu pilih Hapus, lalu konfirmasikan Hapus.

   CloudFormation memulai penghapusan tumpukan dan semua sumber daya yang disertakan.

CloudFormation rincian template

Sumber daya

CloudFormation Template untuk tutorial ini akan membuat sumber daya berikut di akun Anda:

- `AWS::IAM::Role`: Peran IAM federasi yang dapat diasumsikan oleh pengguna yang diautentikasi melalui SAMP iDP Anda.

Konfigurasi

Template mencakup parameter yang dapat dikonfigurasi berikut:

- RoleName- Nama Peran IAM (biarkan kosong untuk nama yang dihasilkan secara otomatis)
- SamlProviderARN - ARN dari SAMP iDP (wajib)
- RoleSessionDuration- Durasi sesi maksimum dalam hitungan detik (3600-43200, default 7200)
- RolePermissionsBoundary- ARN opsional kebijakan batas izin
- RolePath- Jalur untuk peran IAM (default/)
- ManagedPolicy1-5- ARN opsional hingga 5 kebijakan terkelola untuk dilampirkan

CloudFormation template

Simpan kode JSON atau YAMAL berikut sebagai file terpisah untuk digunakan sebagai CloudFormation template untuk tutorial ini.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-federated-role",
  "Parameters": {
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-"
    },
    "SAMLProviderARN": {
      "Type": "String",
      "Description": "ARN of the SAML Identity Provider",
      "AllowedPattern": "^arn:aws:iam::\\d{12}:saml-provider/[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be a valid SAML provider ARN"
```

```
    },
    "RoleSessionDuration": {
      "Type": "Number",
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM role (must start and end with /)",
      "Default": "/",
      "AllowedPattern": "^\/.*\/$|^\/$",
      "ConstraintDescription": "Role path must start and end with forward slash (/)"
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
```

```
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
    "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
    "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"}, ""]}]}
  },
  "Resources": {
    "SAMLFederatedRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
        "Description": "IAM role with SAML provider trust",
        "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
        "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
        "Path": {"Ref": "RolePath"},
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProviderARN"}
              },
              "Action": "sts:AssumeRoleWithSAML",
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
```

```
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
 "AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-federated-role'
```

```
Parameters:
  RoleName:
    Type: String
    Description: 'Name of the IAM Role (leave empty for auto-generated name like
 ''{StackName}-{UniqueId}'')'
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: 'Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-'

  SAMLProviderARN:
    Type: String
    Description: 'ARN of the SAML Identity Provider'
    AllowedPattern: '^arn:aws:iam::\d{12}:saml-provider/[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be a valid SAML provider ARN'

  RoleSessionDuration:
    Type: Number
    Description: 'The maximum session duration (in seconds) that you want to set for
 the specified role (3600-43200)'
    MinValue: 3600
    MaxValue: 43200
    Default: 7200

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
 none)
    Default: ""

  RolePath:
    Type: String
    Description: 'Path for the IAM role (must start and end with /)'
    Default: "/"
    AllowedPattern: '^\/.*\/$|^\/$'
    ConstraintDescription: 'Role path must start and end with forward slash (/)'

  RoleManagedPolicy1:
    Type: String
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
```

```
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]

Resources:
  SAMLFederatedRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Description: 'IAM role with SAML provider trust'
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProviderARN
```

```
            Action: 'sts:AssumeRoleWithSAML'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns:
        !Split
          - ','
          - !Join
            - ','
            - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref 'AWS::NoValue']
              - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref 'AWS::NoValue']
              - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref 'AWS::NoValue']
              - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref 'AWS::NoValue']
              - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref 'AWS::NoValue']

Outputs:
  RoleARN:
    Description: 'ARN of the created IAM Role'
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'
```

Tutorial IAM: Gunakan a CloudFormation template untuk membuat SAML Identity Provider (IdP) dan peran IAM federasi SAML

Untuk membiasakan diri dengan federasi SAML dan kemampuannya, Anda akan menggunakan CloudFormation template untuk mengatur SAML Identity Provider (IdP) dan peran IAM federasi terkait. Tutorial ini menunjukkan cara membuat kedua sumber daya bersama-sama dalam satu tumpukan.

Template membuat IdP SAML yang dapat digunakan untuk akses federasi ke AWS sumber daya, bersama dengan peran IAM yang mempercayai penyedia SAML. Pengguna yang diautentikasi oleh IdP eksternal Anda dapat mengambil peran ini untuk mengakses AWS sumber daya.

Sumber daya yang digunakan terdiri dari yang berikut:

- IdP SAML dikonfigurasi dengan dokumen metadata IdP Anda.
- Peran IAM federasi yang mempercayai IdP SAML dan dapat diasumsikan oleh pengguna yang diautentikasi.

- Kebijakan terkelola yang dapat dikonfigurasi yang dapat dilampirkan ke peran untuk memberikan izin tertentu.

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah melakukan hal berikut:

- Python 3.6 atau yang lebih baru diinstal pada mesin lokal Anda untuk menjalankan perintah Python yang digunakan dalam tutorial ini untuk memformat file XML metadata SAML IdP Anda.
- Dokumen metadata SAML dari IdP eksternal Anda disimpan sebagai file XML.

Buat IdP dan peran SAML menggunakan CloudFormation

Untuk membuat SAML IdP dan peran federasi, Anda akan membuat CloudFormation template dan menggunakannya untuk membuat tumpukan yang berisi kedua sumber daya.

Buattemplat

Pertama, buat CloudFormation template.

1. Di Templat bagian tersebut, klik ikon salin pada tab JSON atau YAML untuk menyalin konten template.
2. Tempelkan isi template ke file baru.
3. Simpan file secara lokal.

Buat tumpukan

Selanjutnya, gunakan template yang telah Anda simpan untuk menyediakan tumpu CloudFormation kan.

1. Buka CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Pada halaman Stacks, dari menu Buat tumpukan, pilih dengan sumber daya baru (standar).
3. Tentukan template:
   a. Di bawah Prasyarat, pilih Pilih template yang ada.
   b. Di bawah Tentukan template, pilih Unggah file template.

c. Pilih Pilih file, navigasikan ke file template, dan pilih.

d. Pilih Berikutnya.

4. Tentukan detail tumpukan berikut:

a. Masukkan nama tumpukan.

b. Untuk IdentityProviderName, Anda dapat membiarkan ini kosong untuk membuat nama secara otomatis berdasarkan nama tumpukan, atau memasukkan nama khusus untuk IdP SAML Anda.

   Contoh: `CompanyIdP` atau `EnterpriseSSO`

c. Untuk IdentityProviderSAMLMetadataDocument, Anda perlu memformat file XML metadata SAML Anda sebagai satu baris sebelum menempelkannya ke bidang ini. Ini diperlukan karena CloudFormation konsol memerlukan konten XML untuk diformat sebagai satu baris ketika melewati parameter konsol.

   Gunakan perintah Python berikut untuk memformat ulang file XML Anda:

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-metadata.xml
```

> **Note**
>
> Dokumen metadata SAML IdP harus diformat sebagai satu baris untuk input parameter konsol. Perintah Python menghapus jeda baris dan spasi putih ekstra untuk membuat format yang diperlukan sambil mempertahankan semua konten dan struktur asli.

   Salin output dari perintah Python dan tempelkan ke IdentityProviderSAMLMetadataDocument bidang.

   Contoh dokumen metadata SAML yang diformat (disingkat):

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
```

```
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Untuk RoleName, Anda dapat membiarkan ini kosong untuk membuat nama secara otomatis berdasarkan nama tumpukan, atau memasukkan nama khusus untuk peran IAM federasi.

   Contoh: `SAML-Developer-Access` atau `SAML-ReadOnly-Role`

e. Untuk parameter lain, terima nilai default atau masukkan sendiri berdasarkan kebutuhan Anda:

   - IdentityProviderAddPrivateKey- Kunci pribadi opsional untuk mendekripsi pernyataan SAML
   - IdentityProviderAssertionEncryptionMode- Mode enkripsi untuk pernyataan SAML

     Contoh nilai:`Allowed`,`Required`, atau biarkan kosong tanpa enkripsi
   - RoleSessionDuration- Durasi sesi maksimum dalam detik (3600-43200, default 7200)

     Contoh: `14400` (4 jam)
   - RolePermissionsBoundary- ARN opsional dari kebijakan batas izin

     Contoh: `arn:aws:iam::123456789012:policy/DeveloperBoundary`
   - RolePath- Jalur untuk peran IAM (default adalah/)

     Contoh: `/saml-roles/`
   - RoleManagedPolicy1-5- ARN opsional hingga 5 kebijakan terkelola untuk dilampirkan

     Contoh untuk RoleManagedPolicy1: `arn:aws:iam::aws:policy/ReadOnlyAccess`

Contoh untuk RoleManagedPolicy2: `arn:aws:iam::123456789012:policy/CustomPolicy`

f. Pilih Berikutnya.

5. Konfigurasikan opsi tumpukan:

a. Di bawah Opsi kegagalan tumpukan, pilih Hapus semua sumber daya yang baru dibuat.

> **Note**
>
> Memilih opsi ini mencegah Anda dari kemungkinan ditagih untuk sumber daya yang kebijakan penghapusannya menentukan bahwa sumber daya tersebut akan dipertahankan meskipun pembuatan tumpukan gagal.

b. Terima semua nilai default lainnya.

c. Di bawah Kemampuan, centang kotak untuk mengetahui bahwa CloudFormation mungkin membuat sumber daya IAM di akun Anda.

d. Pilih Berikutnya.

6. Tinjau detail tumpukan dan pilih Kirim.

CloudFormation membuat tumpukan. Setelah pembuatan tumpukan selesai, sumber daya tumpukan siap digunakan. Anda dapat menggunakan tab Sumber daya pada halaman detail tumpukan untuk melihat sumber daya yang disediakan di akun Anda.

Tumpukan akan menampilkan nilai-nilai berikut, yang dapat Anda lihat di tab Output:

- roLearn: ARN dari peran IAM yang dibuat (misalnya, `arn:aws:iam::123456789012:role/SAML-Developer-Access` atau `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` jika menggunakan nama yang dibuat secara otomatis).
- IdentityProviderARN: ARN dari IdP SAML yang dibuat (misalnya,`arn:aws:iam::123456789012:saml-provider/CompanyIdP`).

Anda akan memerlukan kedua ARN ini saat mengonfigurasi IdP Anda untuk mengirim atribut SAML yang sesuai untuk asumsi peran.

Uji federasi SAML

Setelah SAML IdP dan peran federasi telah dibuat, Anda dapat menguji pengaturan federasi.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Penyedia identitas.

   Anda akan melihat IdP SAML yang baru dibuat dalam daftar.
3. Pilih nama IdP untuk melihat detailnya.

   Pada halaman detail IdP, Anda dapat melihat dokumen metadata SAML dan detail konfigurasi lainnya.
4. Di panel navigasi, pilih Peran.
5. Temukan dan pilih peran federasi yang baru dibuat.

   Pada halaman detail peran, Anda dapat melihat kebijakan kepercayaan yang memungkinkan IdP SAML untuk mengambil peran ini.
6. Pilih tab Relasi kepercayaan untuk meninjau kebijakan kepercayaan.

   Kebijakan kepercayaan harus menunjukkan bahwa IdP SAML dipercaya untuk mengambil peran ini dengan kondisi bahwa pemirsa SAML () `SAML:aud` cocok. `https://signin.aws.amazon.com/saml`

Bersihkan: hapus sumber daya

Sebagai langkah terakhir, Anda akan menghapus tumpukan dan sumber daya yang dikandungnya.

1. Buka CloudFormation konsol.
2. Pada halaman Stacks, pilih tumpukan yang dibuat dari template, lalu pilih Hapus, lalu kon firmasikan H apus.

   CloudFormation memulai penghapusan tumpukan dan semua sumber daya yang disertakan.

CloudFormation rincian template

Sumber daya

Tem CloudFormation plate untuk tutorial ini akan membuat sumber daya berikut di akun Anda:

- `AWS::IAM::SAMLProvider`: IdP SAML yang membangun kepercayaan antara AWS dan IdP eksternal Anda.
- `AWS::IAM::Role`: Peran IAM federasi yang dapat diasumsikan oleh pengguna yang diautentikasi melalui SAML IdP.

Konfigurasi

Template mencakup parameter yang dapat dikonfigurasi berikut:

- IdentityProviderName- Nama IdP SAML (biarkan kosong untuk nama yang dibuat secara otomatis)
- IdentityProviderSAMLMetadataDocument- Dokumen metadata SAML dari IdP Anda (wajib)
- IdentityProviderAddPrivateKey- Kunci pribadi opsional untuk mendekripsi pernyataan SAML
- IdentityProviderAssertionEncryptionMode- Mode enkripsi untuk pernyataan SAML
- RoleName- Nama Peran IAM (biarkan kosong untuk nama yang dibuat secara otomatis)
- RolePath- Jalur untuk peran IAM (default/)
- RolePermissionsBoundary- ARN opsional kebijakan batas izin
- RoleSessionDuration- Durasi sesi maksimum dalam detik (3600-43200, default 7200)
- RoleManagedPolicy1-5- ARN opsional hingga 5 kebijakan terkelola untuk dilampirkan

CloudFormation templat

Simpan kode JSON atau YAML berikut sebagai file terpisah untuk digunakan sebagai CloudFormation template untuk tutorial ini.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp-and-federated-role",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
```

```
      "ConstraintDescription": "Must be empty or contain only alphanumeric
characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    },
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM Role",
      "AllowedPattern": "(^\\/$)|(^\\/.*\\/$)",
      "Default": "/"
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RoleSessionDuration": {
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
```

```
      "Type": "Number",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomProviderName": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderName"}, ""]}]},
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
```

```
   "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
   "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"},
""]}]},
   "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
   "HasAssertionEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]}
 },
 "Resources": {
   "SAMLProvider": {
     "Type": "AWS::IAM::SAMLProvider",
     "Properties": {
       "Name": {"Fn::If": ["HasCustomProviderName", {"Ref":
"IdentityProviderName"}, {"Ref": "AWS::NoValue"}]},
       "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
       "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
"IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
       "AssertionEncryptionMode": {"Fn::If": ["HasAssertionEncryptionMode", {"Ref":
"IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
     }
   },
   "SAMLFederatedRole": {
     "Type": "AWS::IAM::Role",
     "Properties": {
       "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
       "Path": {"Ref": "RolePath"},
       "Description": "SAML federated IAM role for SSO access with specified
permissions",
       "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
       "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
       "AssumeRolePolicyDocument": {
         "Version": "2012-10-17",
         "Statement": [
           {
             "Effect": "Allow",
             "Principal": {
               "Federated": {"Ref": "SAMLProvider"}
             },
             "Action": [
               "sts:AssumeRole",
               "sts:SetSourceIdentity",
```

```
                "sts:TagSession"
              ],
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
"AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    },
    "IdentityProviderARN": {
```

```
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-IdentityProviderARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp-and-federated-role'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens

  IdentityProviderSAMLMetadataDocument:
    Type: String
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"
```

```
 RoleName:
   Type: String
   Description: Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')
   Default: ""
   AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
   ConstraintDescription: "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"

 RolePath:
   Type: String
   Description: Path for the IAM Role
   AllowedPattern: (^\/$)|(^\/.*\/$)
   Default: "/"

 RolePermissionsBoundary:
   Type: String
   Description: Optional ARN of the permissions boundary policy (leave empty for
none)
   Default: ""

 RoleSessionDuration:
   Description: The maximum session duration (in seconds) that you want to set for
the specified role (3600-43200)
   Type: Number
   MinValue: 3600
   MaxValue: 43200
   Default: 7200

 RoleManagedPolicy1:
   Type: String
   Description: Optional managed policy ARN 1
   Default: ""
 RoleManagedPolicy2:
   Type: String
   Description: Optional managed policy ARN 2
   Default: ""
 RoleManagedPolicy3:
   Type: String
   Description: Optional managed policy ARN 3
   Default: ""
 RoleManagedPolicy4:
   Type: String
```

```
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomProviderName: !Not [!Equals [!Ref IdentityProviderName, ""]]
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasAssertionEncryptionMode: !Not [!Equals [!Ref
 IdentityProviderAssertionEncryptionMode, ""]]

Resources:
  SAMLProvider:
    Type: AWS::IAM::SAMLProvider
    Properties:
      Name: !If
        - HasCustomProviderName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasAssertionEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

  SAMLFederatedRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
```

```
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      Description: "SAML federated IAM role for SSO access with specified
 permissions"
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProvider
            Action:
              - 'sts:AssumeRole'
              - 'sts:SetSourceIdentity'
              - 'sts:TagSession'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns: !Split
        - ','
        - !Join
          - ','
          - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref "AWS::NoValue"]
            - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref "AWS::NoValue"]
            - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref "AWS::NoValue"]
            - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref "AWS::NoValue"]
            - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref "AWS::NoValue"]

Outputs:
  RoleARN:
    Description: ARN of the created IAM Role
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'

  IdentityProviderARN:
    Description: ARN of the created SAML Identity Provider
    Value: !Ref SAMLProvider
    Export:
```

```
      Name: !Sub '${AWS::StackName}-IdentityProviderARN'
```

Identitas IAM

Identitas IAM dapat dikaitkan dengan satu atau lebih kebijakan, yang menentukan tindakan apa yang diotorisasi untuk dilakukan identitas, pada AWS sumber daya mana, dan dalam kondisi apa. Identitas IAM termasuk pengguna IAM, grup IAM, dan peran IAM. Entitas IAM adalah jenis identitas yang mewakili pengguna manusia atau beban kerja terprogram yang dapat diautentikasi dan kemudian diberi wewenang untuk melakukan tindakan di. Akun AWS Entitas IAM mencakup pengguna IAM dan peran IAM. Untuk definisi istilah yang umum digunakan, lihatIstilah.

Anda dapat menyatukan identitas yang ada dari penyedia identitas eksternal. Identitas ini akan mengambil peran IAM untuk mengakses AWS sumber daya. Untuk informasi selengkapnya, lihat the section called “Penyedia identitas dan federasi ke AWS”.

Anda juga dapat menggunakan AWS IAM Identity Center untuk membuat dan mengelola identitas dan akses ke AWS sumber daya. Kumpulan izin Pusat Identitas IAM secara otomatis membuat peran IAM yang diperlukan untuk menyediakan akses ke sumber daya. Untuk informasi selengkapnya, lihat Apa itu Pusat Identitas IAM?

Anda dapat menggunakan manajer akses akun — fitur IAM yang memungkinkan Anda menetapkan peran IAM yang ada dengan fitur peran IAM lengkap yang disetel ke pengguna dan grup Pusat Identitas IAM. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.

Ini Pengguna root akun AWS adalah Akun AWS prinsip yang dibuat ketika Anda Akun AWS didirikan. Pengguna root memiliki akses ke semua AWS layanan dan sumber daya di akun. Untuk informasi selengkapnya, lihat the section called “Pengguna root IAM”.

> **Note**
>
> - Ikuti praktik terbaik Keamanan di IAM saat bekerja dengan identitas IAM.
> - Ikuti praktik terbaik pengguna root untuk Anda Akun AWS saat bekerja dengan pengguna root.
> - Jika Anda mengalami masalah saat masuk, lihat Masuk ke Konsol Manajemen AWS.

Pengguna root IAM

Saat pertama kali membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root. Untuk informasi selengkapnya, lihat pengguna root AWS akun.

Pengguna IAM:

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Untuk informasi selengkapnya, lihat pengguna https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html IAM.

Grup pengguna IAM

Grup pengguna IAM adalah identitas yang menentukan kumpulan pengguna IAM. Untuk informasi selengkapnya, lihat Grup pengguna.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin tertentu. Ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Untuk informasi selengkapnya, lihat peran https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html IAM.

Pengguna root akun AWS

Saat pertama kali membuat akun Amazon Web Services (AWS), Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua AWS layanan dan sumber daya di akun. Identitas ini disebut pengguna root AWS akun. Alamat email dan kata sandi yang Anda gunakan untuk membuat Akun AWS adalah kredensil yang Anda gunakan untuk masuk sebagai pengguna root Anda.

- Gunakan pengguna root hanya untuk melakukan tugas yang memerlukan izin tingkat root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.
- Ikuti praktik terbaik pengguna root untuk Anda Akun AWS.
- Jika Anda mengalami masalah saat masuk, lihat Masuk ke Konsol Manajemen AWS.

> **Important**
>
> Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari Anda dan Anda mengikuti praktik terbaik pengguna root untuk Anda Akun AWS. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.

Meskipun MFA diberlakukan untuk pengguna root secara default, MFA memerlukan tindakan pelanggan untuk menambahkan MFA selama pembuatan akun awal atau seperti yang diminta saat masuk. Untuk informasi selengkapnya tentang penggunaan MFA untuk melindungi pengguna root, lihat. Multi-factor otentikasi untuk Pengguna root akun AWS

Selain akses konsol, Anda dapat mengelola akses terprogram untuk pengguna root. Anda bisa Buat kunci akses untuk pengguna root dan Hapus kunci akses untuk pengguna root ketika mereka tidak lagi dibutuhkan.

Mengelola akses root secara terpusat untuk akun anggota

Untuk membantu Anda mengelola kredensi dalam skala besar, Anda dapat mengamankan akses ke kredenial pengguna root untuk akun anggota secara terpusat. AWS Organizations Ketika Anda mengaktifkan AWS Organizations, Anda menggabungkan semua AWS akun Anda ke dalam organisasi untuk manajemen pusat. Memusatkan akses root memungkinkan Anda menghapus kredensi pengguna root dan melakukan tugas-tugas istimewa berikut pada akun anggota.

Hapus kredensi pengguna root akun anggota

Setelah Anda memusatkan akses root untuk akun anggota, Anda dapat memilih untuk menghapus kredensyal pengguna root dari akun anggota di Organizations Anda. Anda dapat menghapus kata sandi pengguna root, kunci akses, menandatangani sertifikat, dan menonaktifkan otentikasi multi-faktor (MFA). Akun baru yang Anda buat di Organizations tidak memiliki kredensi pengguna root secara default. Akun anggota tidak dapat masuk ke pengguna root mereka atau melakukan pemulihan kata sandi untuk pengguna root mereka kecuali pemulihan akun diaktifkan.

Lakukan tugas istimewa yang memerlukan kredensi pengguna root

Beberapa tugas hanya dapat dilakukan ketika Anda masuk sebagai pengguna root akun. Beberapa di antaranya Tugas yang memerlukan kredensial pengguna root dapat dilakukan oleh

akun manajemen atau administrator yang didelegasikan untuk IAM. Untuk mempelajari lebih lanjut tentang mengambil tindakan istimewa pada akun anggota, lihatMelakukan tugas istimewa.

Aktifkan pemulihan akun pengguna root

Jika Anda perlu memulihkan kredensi pengguna root untuk akun anggota, akun manajemen Organisasi atau administrator yang didelegasikan dapat melakukan tugas istimewa Izinkan pemulihan kata sandi. Orang yang memiliki akses ke kotak masuk email pengguna root untuk akun anggota dapat mengatur ulang kata sandi pengguna root untuk memulihkan kredensyal pengguna root. Sebaiknya hapus kredensi pengguna root setelah Anda menyelesaikan tugas yang memerlukan akses ke pengguna root.

Memusatkan akses root untuk akun anggota

KredenSIAL pengguna root adalah kredenSIAL awal Akun AWS yang ditetapkan untuk masing-masing yang memiliki akses lengkap ke semua AWS layanan dan sumber daya di akun. Saat Anda mengaktifkan AWS Organizations, Anda menggabungkan semua AWS akun Anda ke dalam organisasi untuk manajemen pusat. Setiap akun anggota memiliki pengguna root sendiri dengan izin default untuk melakukan tindakan apa pun di akun anggota. Sebaiknya Anda mengamankan secara terpusat kredensi pengguna root dari Akun AWS penggunaan terkelola AWS Organizations untuk mencegah pemulihan kredensia pengguna root dan akses dalam skala besar.

Setelah memusatkan akses root, Anda dapat memilih untuk menghapus kredentif pengguna root dari akun anggota di organisasi Anda. Anda dapat menghapus kata sandi pengguna root, kunci akses, sertifikat penandatanganan, dan menonaktifkan otentikasi multi-faktor (MFA). Akun baru yang Anda buat tidak AWS Organizations memiliki kredenSIAL pengguna root secara default. Akun anggota tidak dapat masuk ke pengguna root mereka atau melakukan pemulihan kata sandi untuk pengguna root mereka.

> **Note**
>
> Meskipun beberapa Tugas yang memerlukan kredensial pengguna root dapat dilakukan oleh akun manajemen atau administrator yang didelegasikan untuk IAM, beberapa tugas hanya dapat dilakukan saat Anda masuk sebagai pengguna root akun.
> Jika Anda perlu memulihkan kredentif pengguna root untuk akun anggota untuk melakukan salah satu tugas ini, ikuti langkah-langkah di Melakukan tugas istimewa dan pilih Iz inkan pemulihan kata sandi. Orang dengan akses ke kotak masuk email pengguna root untuk akun anggota kemudian dapat mengikuti langkah-langkah untuk mengatur ulang kata sandi pengguna root dan masuk ke pengguna root akun anggota.

Sebaiknya hapus kredentif pengguna root setelah Anda menyelesaikan tugas yang memerlukan akses ke pengguna root.

Prasyarat

Sebelum memusatkan akses root, Anda harus memiliki akun yang dikonfigurasi dengan pengaturan berikut:

- Anda harus memiliki izin IAM berikut:
  - `iam:GetAccessKeyLastUsed`
  - `iam:GetAccountSummary`
  - `iam:GetLoginProfile`
  - `iam:GetUser`
  - `iam:ListAccessKeys`
  - `iam:ListMFADevices`
  - `iam:ListSigningCertificates`
  - `sts:AssumeRoot`

> **Note**
>
> Untuk mengaudit status kredensi pengguna root dari akun anggota, Anda dapat menggunakan kebijakan IAMAuditRootUserCredentials AWS terkelola untuk menguraikan izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota, atau menggunakan kebijakan apa pun dengan akses ke`iam:GetAccountSummary`.
> Untuk menghasilkan laporan informasi kredensi pengguna root, kebijakan lain hanya memerlukan `iam:GetAccountSummary` tindakan untuk menghasilkan output yang sama. Anda juga dapat membuat daftar atau mendapatkan informasi kredensi pengguna root individu, termasuk:
>
> - Apakah ada kata sandi pengguna root
> - Apakah ada kunci akses pengguna root dan kapan terakhir digunakan
> - Apakah pengguna root memiliki sertifikat penandatanganan terkait
> - Perangkat MFA terkait pengguna root
> - Daftar status kredensi pengguna root yang dikonsolidasikan

- Anda harus mengatur mas Akun AWS ukmu AWS Organizations.
- Anda harus memiliki izin berikut untuk mengaktifkan fitur ini di organisasi Anda:
  - `iam:EnableOrganizationsRootCredentialsManagement`
  - `iam:EnableOrganizationsRootSessions`
  - `iam:ListOrganizationsFeatures`
  - `organizations:EnableAwsServiceAccess`
  - `organizations:ListAccountsForParent`
  - `organizations:RegisterDelegatedAdministrator`
- Untuk memastikan fungsionalitas konsol yang optimal, sebaiknya aktifkan izin tambahan berikut:
  - `organizations:DescribeAccount`
  - `organizations:DescribeOrganization`
  - `organizations:ListAWSServiceAccessForOrganization`
  - `organizations:ListDelegatedAdministrators`
  - `organizations:ListOrganizationalUnitsForParent`
  - `organizations:ListParents`
  - `organizations:ListTagsForResource`

Mengaktifkan akses root terpusat (konsol)

Untuk mengaktifkan fitur ini untuk akun anggota di Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Manajemen akses root, lalu pilih Aktifkan.

> **Note**
>
> Jika Anda melihat manajemen akses root dinonaktifkan, aktifkan akses tepercaya untuk AWS Identity and Access Management masuk AWS Organizations. Untuk detailnya, lihat AWS IAM dan AWS Organizations di Panduan AWS Organizations Pengguna.

3. Di bagian Kemampuan untuk mengaktifkan, pilih fitur mana yang akan diaktifkan.

- Pilih Manajemen kredenSIAL root untuk mengizinkan akun manajemen dan administrator yang didelegasikan untuk IAM menghapus kredenSIAL pengguna root untuk akun anggota. Anda harus mengaktifkan tindakan root istimewa di akun anggota agar akun anggota dapat memulihkan kredentif pengguna root setelah dihapus.
- Pilih Tindakan root istimewa di akun anggota untuk mengizinkan akun manajemen dan administrator yang didelegasikan untuk IAM melakukan tugas tertentu yang memerlukan kredentif pengguna root.

4. (Opsional) Masukkan ID akun dari administrator yang didelegasikan yang berwenang untuk mengelola akses pengguna root dan mengambil tindakan istimewa pada akun anggota. Kami merekomendasikan akun yang ditujukan untuk tujuan keamanan atau manajemen.
5. Pilih Aktifkan.

Mengaktifkan akses root terpusat (AWS CLI)

Untuk mengaktifkan akses root terpusat dari AWS Command Line Interface (AWS CLI)

1. Jika Anda belum mengaktifkan akses tepercaya untuk AWS Identity and Access Management in AWS Organizations, gunakan perintah berikut: aws organization enable-aws-service-access.
2. Gunakan perintah berikut untuk mengizinkan akun manajemen dan administrator yang didelegasikan menghapus kredenSIAL pengguna root untuk akun anggota: aws iam en able-organizations-root-credentials-management.
3. Gunakan perintah berikut untuk mengizinkan akun manajemen dan administrator yang didelegasikan untuk melakukan tugas-tugas tertentu yang memerlukan kredenSIAL pengguna root: aws iam en able-organizations-root-sessions.
4. (Opsional) Gunakan perintah berikut untuk mendaftarkan administrator yang didelegasikan: aws organisations register- delegated-administrator.

   Contoh berikut menetapkan akun 111111111111 sebagai administrator yang didelegasikan untuk layanan IAM.

```
aws organizations register-delegated-administrator
--service-principal iam.amazonaws.com
--account-id 111111111111
```

Mengaktifkan akses root terpusat (AWS API)

Untuk mengaktifkan akses root terpusat dari AWS API

1. Jika Anda belum mengaktifkan akses tepercaya untuk AWS Identity and Access Management in AWS Organizations, gunakan perintah berikut: EnableAWSServiceAccess.
2. Gunakan perintah berikut untuk mengizinkan akun manajemen dan administrator yang didelegasikan menghapus kredenSIAL pengguna root untuk akun anggota: EnableOrganizationsRootCredentialsManagement.
3. Gunakan perintah berikut untuk mengizinkan akun manajemen dan administrator yang didelegasikan untuk melakukan tugas-tugas tertentu yang memerlukan kredenSIAL pengguna root: EnableOrganizationsRootSessions
4. (Opsional) Gunakan perintah berikut untuk mendaftarkan administrator yang didelegasikan: RegisterDelegatedAdministrator.

Langkah selanjutnya

Setelah mengamankan kredentif istimewa secara terpusat untuk akun anggota di organisasi Anda, lihat Melakukan tugas istimewa untuk mengambil tindakan istimewa pada akun anggota.

Melakukan tugas istimewa pada AWS Organizations akun anggota

Akun AWS Organizations manajemen atau akun administrator yang didelegasikan untuk IAM dapat melakukan beberapa tugas istimewa pada akun anggota yang seharusnya memerlukan kredentif pengguna root. Dengan akses root terpusat, tugas-tugas ini dilakukan melalui sesi istimewa jangka pendek. Sesi ini memberikan kredentif sementara yang dicakup untuk tindakan istimewa tertentu, tanpa memerlukan login pengguna root pada akun anggota.

Setelah meluncurkan sesi istimewa, Anda dapat menghapus kebijakan bucket Amazon S3 yang salah konfigurasi, menghapus kebijakan antrian Amazon SQS yang salah konfigurasi, menghapus kredentif pengguna root untuk akun anggota, dan mengaktifkan kembali kredenSIAL pengguna root untuk akun anggota.

> **Note**
>
> Untuk menggunakan akses root terpusat, Anda harus masuk melalui akun manajemen atau akun administrator yang didelegasikan sebagai pengguna atau peran IAM dengan

`sts:AssumeRoot` izin yang diberikan secara eksplisit. Anda tidak dapat menggunakan kredentif pengguna root untuk meneleponsts:AssumeRoot.

Prasyarat

Sebelum Anda dapat meluncurkan sesi istimewa, Anda harus memiliki pengaturan berikut:

- Anda telah mengaktifkan akses root terpusat di organisasi Anda. Untuk langkah-langkah untuk mengaktifkan fitur ini, lihatMemusatkan akses root untuk akun anggota.
- Akun manajemen atau akun administrator yang didelegasikan memiliki izin berikut: `sts:AssumeRoot`

Mengambil tindakan istimewa pada akun anggota (konsol)

Untuk meluncurkan sesi untuk tindakan istimewa di akun anggota di Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Manajemen akses root.
3. Pilih nama dari daftar akun anggota, dan pilih Ambil tindakan istimewa.
4. Pilih tindakan istimewa yang ingin Anda lakukan di akun anggota.
   - Pilih Hapus kebijakan bucket Amazon S3 untuk menghapus kebijakan bucket yang salah konfigurasi yang menolak semua prinsipal mengakses bucket Amazon S3.
     a. Pilih Browse S3 untuk memilih nama dari bucket yang dimiliki oleh akun anggota, dan pilih Pilih.
     b. Pilih Hapus kebijakan bucket.
     c. Gunakan konsol Amazon S3 untuk memperbaiki kebijakan bucket setelah menghapus kebijakan yang salah dikonfigurasi. Untuk informasi selengkapnya, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3 di Panduan Pengguna Amazon S3.
   - Pilih Hapus kebijakan Amazon SQS untuk menghapus kebijakan berbasis sumber daya Amazon Simple Queue Service yang menolak semua prinsipal mengakses antrian Amazon SQS.

a. Masukkan nama antrian dalam nama antrian SQS, dan pilih H apus kebijakan SQS.

b. Gunakan konsol Amazon SQS untuk memperbaiki kebijakan antrian setelah menghapus kebijakan yang salah dikonfigurasi. Untuk informasi selengkapnya, lihat Mengonfigurasi kebijakan akses di Amazon SQS di Panduan Pengembang Amazon SQS.

- Pilih Hapus kredenSIAL root untuk menghapus akses root dari akun anggota. Menghapus kredenSIAL pengguna root akan menghapus kata sandi pengguna root, kunci akses, sertifikat penandatanganan, dan menonaktifkan otentikasi multi-faktor (MFA) untuk akun anggota.

  - Pilih H apus kredenSIAL root.

- Pilih Izinkan pemulihan kata sandi untuk memulihkan kredenSIAL pengguna root untuk akun anggota.

  Opsi ini hanya tersedia jika akun anggota tidak memiliki kredenSIAL pengguna root.

  a. Pilih Iz inkan pemulihan kata sandi.

  b. Setelah mengambil tindakan istimewa ini, orang dengan akses ke kotak masuk email pengguna root untuk akun anggota dapat mengatur ulang kata sandi pengguna root dan masuk ke pengguna root akun anggota.

Mengambil tindakan istimewa pada akun anggota (AWS CLI)

Untuk meluncurkan sesi untuk tindakan istimewa di akun anggota dari AWS Command Line Interface

1. Gunakan perintah berikut untuk mengasumsikan sesi pengguna root: aws sts assume- root.

> **Note**
>
> Titik akhir global tidak didukung untuk`sts:AssumeRoot`. Anda harus mengirim permintaan ini ke titik AWS STS akhir Regional. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Saat Anda meluncurkan sesi pengguna root yang memiliki hak istimewa untuk akun anggota, Anda harus menentukan `task-policy-arn` untuk menjangkau sesi ke tindakan istimewa yang

akan dilakukan selama sesi. Anda dapat menggunakan salah satu kebijakan ter AWS kelola berikut untuk menjangkau tindakan sesi istimewa.

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Untuk membatasi tindakan yang dapat dilakukan akun manajemen atau administrator yang didelegasikan selama sesi pengguna root yang memiliki hak istimewa, Anda dapat menggunakan kunci AWS STS kondisi sts: TaskPolicyArn.

Dalam contoh berikut, administrator yang didelegasikan mengasumsikan root untuk menghapus kredenSIAL pengguna root untuk ID akun anggota*111122223333*.

```
aws sts assume-root \
  --target-principal 111122223333 \
  --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/
IAMDeleteRootUserCredentials \
  --duration-seconds 900
```

2. Gunakan`SessionToken`,`AccessKeyId`, dan `SecretAccessKey` dari respons untuk melakukan tindakan istimewa di akun anggota. Anda dapat menghilangkan nama pengguna dan kata sandi dalam permintaan untuk default ke akun anggota.
   - Periksa status kredenSIAL pengguna root. Gunakan perintah berikut untuk memeriksa status kredenSIAL pengguna root untuk akun anggota.
     - dapatkan-pengguna
     - get-login-profil
     - kunci-akses daftar
     - daftar sertifikat penandatanganan
     - daftar-mfa-perangkat
     - dapatkan-akses-kunci-terakhir digunakan
   - Hapus kredenSIAL pengguna root. Gunakan perintah berikut untuk menghapus akses root. Anda dapat menghapus kata sandi pengguna root, kunci akses, sertifikat penandatanganan,

dan menonaktifkan otentikasi multi-faktor (MFA) untuk menghapus semua akses ke dan pemulihan pengguna root.

- hapus login-profil
- hapus kunci akses
- hapus sertifikat penandatanganan
- nonaktifkan-perangkat mfa

- Hapus kebijakan bucket Amazon S3. Gunakan perintah berikut untuk membaca, mengedit, dan menghapus kebijakan bucket yang salah konfigurasi yang menolak semua prinsipal mengakses bucket Amazon S3.
  - daftar bucket
  - kebijakan get-bucket
  - kebijakan put-bucket
  - kebijakan hapus bucket
- Hapus kebijakan Amazon SQS. Gunakan perintah berikut untuk melihat dan menghapus kebijakan berbasis sumber daya Amazon Simple Queue Service yang menolak semua prinsipal mengakses antrian Amazon SQS.
  - daftar antrian
  - dapatkan antri-url
  - get-queue-atribut
  - atribut-set-queue-
- Izinkan pemulihan kata sandi. Gunakan perintah berikut untuk melihat nama pengguna dan memulihkan kredenSIAL pengguna root untuk akun anggota.
  - get-login-profil
  - buat-login-profil

Mengambil tindakan istimewa pada akun anggota (AWS API)

Untuk meluncurkan sesi untuk tindakan istimewa di akun anggota dari AWS API

1. Gunakan perintah berikut untuk mengasumsikan sesi pengguna root: AssumeRoot.

> **Note**
>
> Titik akhir global tidak didukung untuk AssumeRoot. Anda harus mengirim permintaan ini ke titik AWS STS akhir Regional. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Saat Anda meluncurkan sesi pengguna root yang memiliki hak istimewa untuk akun anggota, Anda harus menentukan `TaskPolicyArn` untuk menjangkau sesi ke tindakan istimewa yang akan dilakukan selama sesi. Anda dapat menggunakan salah satu kebijakan ter AWS kelola berikut untuk menjangkau tindakan sesi istimewa.

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Untuk membatasi tindakan yang dapat dilakukan akun manajemen atau administrator yang didelegasikan selama sesi pengguna root yang memiliki hak istimewa, Anda dapat menggunakan kunci AWS STS kondisi sts: TaskPolicyArn.

Dalam contoh berikut, administrator yang didelegasikan mengasumsikan root untuk membaca, mengedit, dan menghapus kebijakan berbasis sumber daya yang salah dikonfigurasi untuk bucket Amazon S3 untuk ID akun anggota. *111122223333*

```
https://sts.us-east-2.amazonaws.com/
  ?Version=2011-06-15
  &Action=AssumeRoot
  &TargetPrincipal=111122223333
  &PolicyArns.arn=arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy
  &DurationSeconds=900
```

2. Gunakan`SessionToken`,`AccessKeyId`, dan `SecretAccessKey` dari respons untuk melakukan tindakan istimewa di akun anggota. Anda dapat menghilangkan nama pengguna dan kata sandi dalam permintaan untuk default ke akun anggota.

- Periksa status kredenSIAL pengguna root. Gunakan perintah berikut untuk memeriksa status kredenSIAL pengguna root untuk akun anggota.
  - GetUser
  - GetLoginProfile
  - ListAccessKeys
  - ListSigningCertificates
  - ListMfaDevices
  - GetAccessKeyLastUsed
- Hapus kredenSIAL pengguna root. Gunakan perintah berikut untuk menghapus akses root. Anda dapat menghapus kata sandi pengguna root, kunci akses, sertifikat penandatanganan, dan menonaktifkan otentikasi multi-faktor (MFA) untuk menghapus semua akses ke dan pemulihan pengguna root.
  - DeleteLoginProfile
  - DeleteAccessKey
  - DeleteSigningCertificate
  - DeactivateMfaDevice
- Hapus kebijakan bucket Amazon S3. Gunakan perintah berikut untuk membaca, mengedit, dan menghapus kebijakan bucket yang salah konfigurasi yang menolak semua prinsipal mengakses bucket Amazon S3.
  - ListBuckets
  - GetBucketPolicy
  - PutBucketPolicy
  - DeleteBucketPolicy
- Hapus kebijakan Amazon SQS. Gunakan perintah berikut untuk melihat dan menghapus kebijakan berbasis sumber daya Amazon Simple Queue Service yang menolak semua prinsipal mengakses antrian Amazon SQS.
  - ListQueues
  - GetQueueUrl
  - GetQueueAttributes
  - SetQueueAttributes

- Izinkan pemulihan kata sandi. Gunakan perintah berikut untuk melihat nama pengguna dan memulihkan kredenSIAL pengguna root untuk akun anggota.
  - GetLoginProfile
  - CreateLoginProfile

Multi-factor otentikasi untuk Pengguna root akun AWS

> **Important**
>
> AWS merekomendasikan agar Anda menggunakan kunci sandi atau kunci keamanan untuk MFA AWS, sedapat mungkin karena mereka lebih tahan terhadap serangan seperti phishing. Untuk informasi selengkapnya, lihat Kunci sandi dan kunci keamanan.

Multi-factor otentikasi (MFA) adalah mekanisme sederhana dan efektif untuk meningkatkan keamanan Anda. Faktor pertama — kata sandi Anda — adalah rahasia yang Anda hafal, juga dikenal sebagai faktor pengetahuan. Faktor lain dapat berupa faktor kepemilikan (sesuatu yang Anda miliki, seperti kunci keamanan) atau faktor inherensi (sesuatu yang Anda miliki, seperti pemindaian biometrik). Untuk meningkatkan keamanan, kami sangat menyarankan Anda mengonfigurasi otentikasi multi-faktor (MFA) untuk membantu melindungi sumber daya Anda AWS .

> **Note**
>
> Semua Akun AWS jenis (akun mandiri, manajemen, dan anggota) memerlukan MFA untuk dikonfigurasi untuk pengguna root mereka. Pengguna harus mendaftarkan MFA dalam waktu 35 hari sejak upaya masuk pertama mereka untuk mengakses Konsol Manajemen AWS jika MFA belum diaktifkan.

Anda dapat mengaktifkan MFA untuk Pengguna root akun AWS dan pengguna IAM. Saat Anda mengaktifkan MFA untuk pengguna root, itu hanya memengaruhi kredenSIAL pengguna root. Untuk informasi selengkapnya tentang cara mengaktifkan MFA untuk pengguna IAM Anda, lihat. AWS Multi-factor otentikasi di IAM

> **Note**
>
> Akun AWS dikelola menggunakan AWS Organizations mungkin memiliki opsi untuk mengelola akses root secara ter pusat untuk akun anggota untuk mencegah pemulihan kredensia dan akses dalam skala besar. Jika opsi ini diaktifkan, Anda dapat menghapus kredenSIAL pengguna root dari akun anggota, termasuk kata sandi dan MFA, secara efektif mencegah masuk sebagai pengguna root, pemulihan kata sandi, atau pengaturan MFA. Atau, jika Anda lebih suka mempertahankan metode masuk berbasis kata sandi, amankan akun Anda dengan mendaftarkan MFA untuk meningkatkan perlindungan akun.

Sebelum mengaktifkan MFA untuk pengguna root, tinjau dan perbarui pengaturan akun dan informasi kontak untuk memastikan bahwa Anda memiliki akses ke email dan nomor telepon. Jika perangkat MFA Anda hilang, dicuri, atau tidak berfungsi, Anda masih dapat masuk sebagai pengguna akar dengan memverifikasi identitas menggunakan email dan nomor telepon tersebut. Untuk mempelajari tentang proses masuk menggunakan faktor autentikasi alternatif, lihat Memulihkan identitas yang dilindungi MFA di IAM. Untuk menonaktifkan fitur ini, hubungi AWS Dukungan.

AWS mendukung jenis MFA berikut untuk pengguna root Anda:

- Kunci sandi dan kunci keamanan
- Aplikasi otentikator virtual
- Token TOTP perangkat keras

Kunci sandi dan kunci keamanan

AWS Identity and Access Management mendukung kunci sandi dan kunci keamanan untuk MFA. Berdasarkan standar FIDO, kunci sandi menggunakan kriptografi kunci publik untuk memberikan otentikasi yang kuat dan tahan phishing yang lebih aman daripada kata sandi. AWS mendukung dua jenis kunci sandi: kunci sandi terikat perangkat (kunci keamanan) dan kunci sandi yang disinkronkan.

- Kunci keamanan: Ini adalah perangkat fisik, seperti YubiKey, digunakan sebagai faktor kedua untuk otentikasi. Satu kunci keamanan dapat mendukung beberapa akun pengguna root dan pengguna IAM.
- Kunci sandi yang disinkron kan: Ini menggunakan manajer kredensia dari penyedia seperti Google, Apple, akun Microsoft, dan layanan pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua.

Anda dapat menggunakan otentikator biometrik bawaan, seperti Touch ID di Apple MacBooks, untuk membuka kunci pengelola kredensi dan masuk. AWS Kunci sandi dibuat dengan penyedia pilihan Anda menggunakan sidik jari, wajah, atau PIN perangkat Anda. Anda juga dapat menggunakan kunci sandi otentikasi lintas perangkat (CDA) dari satu perangkat, seperti perangkat seluler atau kunci keamanan perangkat keras, untuk masuk di perangkat lain seperti laptop. Untuk informasi selengkapnya, lihat otentikasi lintas perangkat (CDA).

Anda dapat menyinkronkan kunci sandi di seluruh perangkat Anda untuk memfasilitasi login dengan AWS, meningkatkan kegunaan dan pemulihan. Untuk informasi selengkapnya tentang mengaktifkan kunci sandi dan kunci keamanan, lihat. Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Aliansi FIDO menyimpan daftar semua produk Bersertifikat FIDO yang kompatibel dengan spesifikasi FIDO.

Aplikasi otentikator virtual

Aplikasi otentikator virtual berjalan di telepon atau perangkat lain dan meniru perangkat fisik. Aplikasi otentikator virtual menerapkan algoritma kata sandi satu kali berbasis waktu (TOTP) dan mendukung beberapa token pada satu perangkat. Pengguna harus mengetikkan kode yang valid dari perangkat saat diminta saat masuk. Setiap token yang diberikan kepada pengguna harus unik. Pengguna tidak dapat mengetikkan kode dari token pengguna lain untuk diautentikasi.

Kami menyarankan agar Anda menggunakan perangkat MFA virtual saat menunggu untuk mendapatkan persetujuan pembelian perangkat keras atau saat menunggu kedatangan perangkat keras Anda. Untuk daftar beberapa aplikasi yang didukung yang dapat Anda gunakan sebagai perangkat MFA virtual, lihat Oten Multi-Factor tikasi (MFA). Untuk petunjuk tentang cara menyiapkan perangkat MFA virtual dengan AWS, lihatAktifkan perangkat MFA virtual untuk pengguna root (konsol).

Token TOTP perangkat keras

Perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali berbasis waktu (TOTP). Pengguna harus mengetik kode yang valid dari perangkat di halaman web kedua saat masuk. Setiap perangkat MFA yang ditetapkan ke pengguna harus unik. Pengguna tidak dapat mengetik kode dari perangkat pengguna lain untuk diautentikasi. Untuk informasi tentang perangkat MFA perangkat keras yang didukung, lihat Oten Multi-Factor tikasi (MFA). Untuk petunjuk tentang cara menyiapkan token TOTP perangkat keras dengan AWS, lihatAktifkan token TOTP perangkat keras untuk pengguna root (konsol).

Jika Anda ingin menggunakan perangkat MFA fisik, sebaiknya gunakan kunci keamanan FIDO sebagai alternatif perangkat TOTP perangkat keras. Kunci keamanan FIDO menawarkan manfaat tanpa persyaratan baterai, resistensi phishing, dan mendukung beberapa pengguna root dan IAM pada satu perangkat untuk keamanan yang ditingkatkan.

Topik

- Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)
- Aktifkan perangkat MFA virtual untuk pengguna root (konsol)
- Aktifkan token TOTP perangkat keras untuk pengguna root (konsol)

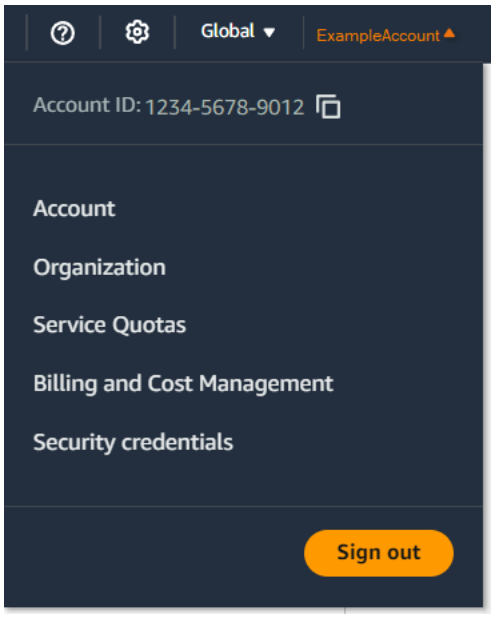
Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Anda dapat mengonfigurasi dan mengaktifkan kunci sandi untuk pengguna root Anda dari Konsol Manajemen AWS satu-satunya, bukan dari AWS CLI atau AWS API.

Untuk mengaktifkan kunci sandi atau kunci keamanan untuk pengguna root Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan.



3. Pada pengguna root Anda Halaman kredensial keamanan saya, di bawah Multi-factor autentikasi (MFA), pilih Tetapkan perangkat MFA.

4. Pada halaman nama perangkat MFA, masukkan nama Perangkat, pilih Kunci Sandi atau Kunci Keamanan, lalu pilih Berikutnya.
5. Pada Siapkan perangkat, atur kunci sandi Anda. Buat passkey dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci keamanan FIDO ke port USB komputer Anda dan mengetuknya.
6. Ikuti petunjuk di browser Anda untuk memilih penyedia kunci sandi atau tempat Anda ingin menyimpan kunci sandi untuk digunakan di seluruh perangkat Anda.
7. Pilih Lanjutkan.

Anda sekarang telah mendaftarkan passkey Anda untuk digunakan dengan AWS. Lain kali Anda menggunakan kredensi pengguna root Anda untuk masuk, Anda harus mengautentikasi dengan kunci sandi Anda untuk menyelesaikan proses masuk.

Untuk membantu memecahkan masalah dengan kunci keamanan FIDO Anda, lihat. Memecahkan Masalah Kunci Sandi dan Kunci Keamanan FIDO

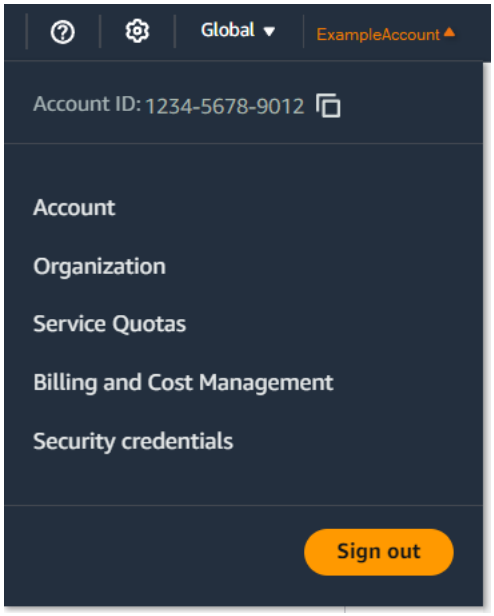
Aktifkan perangkat MFA virtual untuk pengguna root (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk mengkonfigurasi dan mengaktifkan perangkat MFA virtual untuk pengguna root Anda. Untuk mengaktifkan perangkat MFA untuk Akun AWS, Anda harus masuk AWS menggunakan kredensional pengguna root Anda.

Untuk mengonfigurasi dan mengaktifkan perangkat MFA virtual untuk digunakan dengan pengguna akar Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredenSIAL pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sisi kanan bilah navigasi, pilih nama akun Anda, dan pilih Kreden SIAL Keamanan.



3. Di bagian O Multi-Factor tentikasi (MFA), pilih Tet apkan perangkat MFA.
4. Di wizard, ketik nama Perangkat, pilih aplikasi Authenticator, lalu pilih Next.

   IAM menghasilkan dan menampilkan informasi konfigurasi untuk perangkat MFA virtual, termasuk grafik kode QR. Grafik adalah representasi kunci konfigurasi rahasia yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.
5. Buka aplikasi MFA virtual di perangkat.

   Jika aplikasi MFA virtual mendukung beberapa perangkat atau akun MFA virtual, pilih opsi untuk membuat perangkat atau akun MFA virtual baru.
6. Cara termudah untuk mengonfigurasi aplikasi adalah menggunakan aplikasi untuk memindai kode QR. Jika Anda tidak dapat memindai kode, Anda dapat mengetik informasi konfigurasi secara manual. Kode QR dan kunci konfigurasi rahasia yang dihasilkan oleh IAM terkait dengan akun Anda Akun AWS dan tidak dapat digunakan dengan akun yang berbeda. Namun, mereka dapat digunakan kembali untuk mengonfigurasi perangkat MFA baru untuk akun Anda jika Anda kehilangan akses ke perangkat MFA asli.

   - Untuk menggunakan kode QR untuk mengonfigurasi perangkat MFA virtual, dari wizard, pilih Tampilkan kode QR. Lalu, ikuti petunjuk aplikasi untuk memindai kode. Misalnya, Anda mungkin perlu memilih ikon kamera atau memilih perintah seperti Kode batang akun pemindaian, lalu gunakan kamera perangkat untuk memindai kode QR.
   - Di wizard Pengaturan perangkat, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam aplikasi MFA Anda.

> **Important**
>
> Buat cadangan aman kode QR atau kunci konfigurasi rahasia, atau pastikan Anda mengaktifkan beberapa perangkat MFA untuk akun Anda. Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Anda Pengguna root akun AWS dan IAM. Perangkat MFA virtual mungkin tidak tersedia, misalnya, jika Anda kehilangan ponsel cerdas tempat perangkat MFA virtual di-host. Jika itu terjadi dan Anda tidak dapat masuk ke akun Anda tanpa perangkat MFA tambahan yang terpasang pada pengguna atau bahkan olehPemulihan perangkat MFA pengguna akar, Anda tidak akan dapat masuk ke akun Anda dan Anda harus menghubungi layanan pelanggan untuk menghapus perlindungan MFA untuk akun tersebut.

Alat mulai menghasilkan angka enam-digit.

7. Di wizard, di kotak kode MFA 1, ketik kata sandi satu kali yang saat ini muncul di perangkat MFA virtual. Tunggu hingga 30 detik untuk membuat kata sandi sekali pakai yang baru. Lalu ketikkan kata sandi satu kali kedua ke kotak Kode MFA 2. Pilih Tambahkan MFA.

> **Important**
>
> Kirim permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi lebih lanjut tentang penggunaan MFA dengan Konsol Manajemen AWS, lihat MFA mengaktifkan login.

Aktifkan token TOTP perangkat keras untuk pengguna root (konsol)

Anda dapat mengonfigurasi dan mengaktifkan perangkat MFA fisik untuk pengguna root Anda dari Konsol Manajemen AWS satu-satunya, bukan dari AWS CLI atau AWS API.

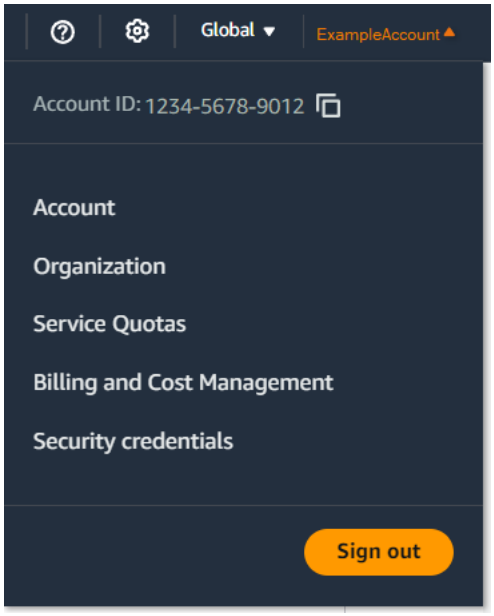
> **Note**
>
> Anda mungkin melihat teks yang berbeda, seperti Masuk menggunakan MFA dan Atasi masalah perangkat autentikasi Anda. Namun, fitur yang sama disediakan. Dalam kedua kasus tersebut, jika Anda tidak dapat memverifikasi alamat email akun dan nomor telepon Anda menggunakan faktor otentikasi alternatif, hubungi AWS Dukunganuntuk menghapus pengaturan MFA Anda.

Untuk mengaktifkan token TOTP perangkat keras untuk pengguna root Anda (konsol)

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.

2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan.



3. Perluas bagian Multi-factor otentikasi (MFA).
4. Pilih Tetapkan perangkat MFA.
5. Di wizard, ketik nama Perangkat, pilih token TOTP Perangkat Keras, lalu pilih Berikutnya.
6. Di kotak Nomor seri, ketik nomor seri yang ditemukan di bagian belakang perangkat MFA.
7. Di kotak Kode MFA 1, ketik angka enam-digit yang ditampilkan oleh perangkat MFA. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.



8. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke kotak Kode MFA 2. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.
9. Pilih Tambahkan MFA. Perangkat MFA sekarang dikaitkan dengan. Akun AWS

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak akan sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat mensinkronisasi ulang perangkat.

Saat berikutnya Anda menggunakan kredensial pengguna akar Anda untuk masuk, Anda harus mengetik kode dari perangkat MFA.

Ubah kata sandi untuk Pengguna root akun AWS

Anda dapat mengubah alamat email dan kata sandi dari halaman Kreden SIAL Keamanan atau Akun. Anda juga dapat memilih L upa kata sandi? pada halaman AWS masuk untuk mengatur ulang kata sandi Anda.

Untuk mengubah kata sandi pengguna root, Anda harus masuk sebagai Pengguna root akun AWS dan bukan sebagai pengguna IAM. Untuk mempelajari cara mengatur ulang kata sandi pengguna akar yang lupa, lihat Setel ulang kata sandi pengguna root yang hilang atau terlupakan.

Untuk melindungi kata sandi Anda, penting untuk mengikuti praktik terbaik ini:

- Ubah kata sandi Anda secara berkala.
- Jaga kerahasiaan kata sandi Anda karena siapa pun yang tahu kata sandi Anda dapat mengakses akun Anda.
- Gunakan kata sandi yang berbeda AWS dari yang Anda gunakan di situs lain.

- Hindari kata sandi yang mudah ditebak. Ini termasuk kata sandi seperti `secret`, `password`, `amazon`, atau `123456`. Hindari juga hal-hal seperti kata-kata kamus, nama Anda, alamat email, atau informasi pribadi lainnya yang dapat diperoleh seseorang dengan mudah.

> **Important**
>
> Akun AWS dikelola menggunakan AWS Organizations mungkin ??? mengaktifkan akses root terpusat untuk akun anggota. Akun anggota ini tidak memiliki kredentif pengguna root, tidak dapat masuk sebagai pengguna root, dan dicegah memulihkan kata sandi pengguna root. Hubungi administrator Anda jika Anda perlu melakukan tugas yang memerlukan kredentif pengguna root.

Konsol Manajemen AWS

Untuk mengubah kata sandi bagi pengguna akar

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai pengguna atau peran IAM.

1. Buka AWS Management Console dan masuk menggunakan kredenSIAL pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Kreden SIAL Keamanan.
3. Pada halaman Akun, di samping Pengaturan akun, pilih Edit. Anda diminta untuk melakukan autentikasi ulang untuk tujuan keamanan.

> **Note**
>
> Jika Anda tidak melihat opsi Edit, kemungkinan Anda tidak masuk sebagai pengguna root untuk akun Anda. Anda tidak dapat mengubah setelan akun saat masuk sebagai pengguna atau peran IAM.

4. Pada halaman Per barui pengaturan akun, di bawah Kata Sandi, pilih Edit.
5. Pada halaman Per barui kata sandi Anda, isi bidang untuk Kata sandi saat ini, Kata sandi baru, dan Kon firmasi kata sandi baru.

> **Important**
>
> Pastikan untuk memilih password yang kuat. Meskipun Anda dapat menetapkan kebijakan kata sandi akun untuk pengguna IAM, kebijakan tersebut tidak berlaku untuk pengguna root.

AWS mengharuskan kata sandi Anda memenuhi persyaratan berikut:

- Itu harus memiliki minimal 8 karakter dan maksimum 128 karakter.
- Ini harus mencakup minimal tiga dari campuran jenis karakter berikut: huruf besar, huruf kecil, angka, dan! @ # $% ^ & * () <> [] {} | _+-= simbol.
- Itu tidak boleh identik dengan Akun AWS nama atau alamat email Anda.

6. Pilih Simpan perubahan.

AWS CLI or AWS SDK

Tugas ini tidak didukung dalam AWS CLI atau oleh operasi API dari salah satu AWS SDK. Anda dapat melakukan tugas ini hanya dengan menggunakan Konsol Manajemen AWS.

Setel ulang kata sandi pengguna root yang hilang atau terlupakan

Ketika Anda pertama kali membuat Akun AWS, Anda memberikan alamat email dan kata sandi. Ini adalah Pengguna root akun AWS kredensimalmu. Jika Anda lupa kata sandi pengguna root Anda, Anda dapat mengatur ulang kata sandi dari Konsol Manajemen AWS.

Akun AWS dikelola menggunakan AWS Organizations mungkin ??? mengaktifkan akses root terpusat untuk akun anggota. Akun anggota ini tidak memiliki kredentif pengguna root, tidak dapat masuk sebagai pengguna root, dan dicegah memulihkan kata sandi pengguna root. Hubungi administrator Anda jika Anda perlu melakukan tugas yang memerlukan kredentif pengguna root.

> **Important**
>
> Kesulitan masuk AWS? Pastikan Anda berada di halaman masuk AWS yang benar untuk jenis pengguna Anda. Jika Anda adalah Pengguna root akun AWS (pemilik akun), Anda dapat masuk AWS menggunakan kredensional yang Anda atur saat membuat Akun AWS. Jika Anda adalah pengguna IAM, administrator akun Anda dapat memberi Anda kredenSIAL yang dapat Anda gunakan untuk AWS masuk. Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini, karena formulir diterima oleh tim AWS Dokumentasi, tidak Dukungan. Sebagai gantinya, pada halaman Hu bungi Kami pilih Masih tidak dapat masuk ke AWS akun Anda dan kemudian pilih salah satu opsi dukungan yang tersedia.

Untuk mengatur ulang kata sandi pengguna root Anda

1. Buka AWS Management Console dan masuk menggunakan kredenSIAL pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.

   > **Note**
   >
   > Jika Anda masuk ke kredensional pengguna Konsol Manajemen AWS dengan IAM, maka Anda harus keluar sebelum Anda dapat mengatur ulang kata sandi pengguna root. Jika Anda melihat halaman masuk pengguna IAM khusus akun, pilih Sign-in menggunakan kredenSIAL akun root di dekat bagian bawah halaman. Jika perlu, berikan alamat email akun Anda dan pilih Selanjutnya untuk mengakses halaman Masuk ke pengguna akar.

2. Pilih Lupa kata sandi Anda?.

> **Note**
>
> Jika Anda pengguna IAM, opsi ini tidak tersedia. L upa kata sandi Anda? opsi hanya tersedia untuk akun pengguna root. Pengguna IAM harus meminta administrator mereka untuk mengatur ulang kata sandi yang terlupakan. Untuk informasi selengkapnya, lihat Saya lupa kata sandi pengguna IAM untuk AWS akun saya. Jika Anda masuk melalui portal AWS akses, lihat Mengatur ulang kata sandi pengguna Pusat Identitas IAM Anda.

3. Masukkan alamat email yang terkait dengan akun tersebut. Kemudian berikan teks CAPTCHA dan pilih Lanjutkan.
4. Periksa email yang terkait dengan Anda Akun AWS untuk pesan dari Amazon Web Services. Email akan datang dari alamat yang diakhiri dengan`@verify.signin.aws`. Ikuti petunjuk di dalam email. Jika Anda tidak melihat email di akun Anda, periksa folder spam Anda. Jika Anda tidak lagi memiliki akses ke email, lihat Saya tidak memiliki akses ke email untuk AWS akun saya di Panduan AWS Sign-In Pengguna.

Buat kunci akses untuk pengguna root

> **Warning**
>
> Kami sangat menyarankan agar Anda tidak membuat pasangan kunci akses untuk pengguna root Anda. Karena hanya beberapa tugas yang memerlukan pengguna root dan Anda biasanya jarang melakukan tugas-tugas tersebut, sebaiknya masuk Konsol Manajemen AWS ke untuk melakukan tugas pengguna root. Sebelum membuat kunci akses, tinjau alternatif untuk kunci akses jangka panjang.

Meskipun kami tidak merekomendasikannya, Anda dapat membuat kunci akses untuk pengguna root sehingga Anda dapat menjalankan perintah di AWS Command Line Interface (AWS CLI) atau menggunakan operasi API dari salah satu AWS SDK menggunakan kredenSIAL pengguna root. Saat Anda membuat access key, Anda membuat access key ID dan secret access key sebagai satu set. Selama pembuatan kunci akses, AWS memberi Anda satu kesempatan untuk melihat dan mengunduh bagian kunci akses rahasia dari kunci akses. Jika Anda tidak mengunduhnya atau Anda kehilangannya, Anda dapat menghapus access key itu lalu membuat yang baru. Anda dapat membuat kunci akses pengguna root dengan konsol, AWS CLI, atau AWS API.

Access key yang baru dibuat memiliki status aktif, yang berarti bahwa Anda dapat menggunakan access key itu untuk panggilan CLI dan API. Anda dapat menetapkan hingga dua kunci akses ke pengguna root.

Kunci akses yang tidak digunakan harus dinonaktifkan. Setelah kunci akses tidak aktif, Anda tidak dapat menggunakannya untuk panggilan API. Kunci yang tidak aktif masih diperhitungkan dalam batas Anda. Anda dapat membuat atau menghapus access key kapan saja. Namun, ketika Anda menghapus access key, kunci tersebut hilang selamanya dan tidak dapat diambil kembali.

Konsol Manajemen AWS

Untuk membuat kunci akses untuk Pengguna root akun AWS

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai pengguna atau peran IAM.

1. Buka AWS Management Console dan masuk menggunakan kredenSIAL pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Kreden SIAL Keamanan.
3. Pada bagian Access key, pilih Buat access key. Jika opsi ini tidak tersedia, maka Anda sudah memiliki jumlah maksimum kunci akses. Anda harus menghapus salah satu kunci akses yang ada sebelum Anda dapat membuat kunci baru. Untuk informasi selengkapnya, lihat Kuota https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_iam-quotas.html#reference_iam-quotas-entities Objek IAM.
4. Pada halaman Alternatif untuk kunci akses pengguna root, tinjau rekomendasi keamanan. Untuk melanjutkan, pilih kotak centang, lalu pilih Buat kunci akses.
5. Pada halaman Ambil kunci akses, ID kunci akses Anda ditampilkan.

6. Di bawah Kunci akses rahasia, pilih Tampilkan lalu salin ID kunci akses dan kunci rahasia dari jendela browser Anda dan tempelkan di tempat yang aman. Atau, Anda dapat memilih Unduh file.csv yang akan mengunduh file bernama `rootkey.csv` yang berisi ID kunci akses dan kunci rahasia. Simpan file di tempat yang aman.
7. Pilih Selesai. Ketika Anda tidak lagi memerlukan kunci akses, kami sarankan Anda menghapusnya, atau setidaknya mempertimbangkan untuk menonaktifkannya sehingga tidak ada yang dapat menyalahgunakannya.

AWS CLI & SDKs

Untuk membuat kunci akses untuk pengguna root

> **Note**
>
> Untuk menjalankan perintah atau operasi API berikut sebagai pengguna root, Anda harus sudah memiliki satu pasangan kunci akses aktif. Jika Anda tidak memiliki kunci akses, buat kunci akses pertama menggunakan Konsol Manajemen AWS. Kemudian, Anda dapat menggunakan kredenSIAL dari kunci akses pertama dengan AWS CLI untuk membuat kunci akses kedua, atau untuk menghapus kunci akses.

- AWS CLI: buat-kunci-akses aws iam

  Example

```
$ aws iam create-access-key
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- AWS API: CreateAccessKey dalam Refer ensi API IAM.

Hapus kunci akses untuk pengguna root

Anda dapat menggunakan Konsol Manajemen AWS, AWS CLI atau AWS API untuk menghapus kunci akses pengguna root.

Konsol Manajemen AWS

Untuk menghapus kunci akses untuk pengguna root

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai pengguna atau peran IAM.

1. Buka AWS Management Console dan masuk menggunakan kredensi pengguna root Anda.

   Untuk petunjuk, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.
2. Di sudut kanan atas konsol, pilih nama atau nomor akun Anda, lalu pilih Kredensial Keamanan.
3. Di bagian Kunci akses, pilih tombol akses yang ingin Anda hapus, lalu, di bawah Tindakan, pilih Hapus.

   > **Note**
   >
   > Atau, Anda dapat Menonaktifkan kunci akses, alih-alih menghapusnya secara permanen. Dengan cara ini Anda dapat melanjutkan menggunakannya di masa depan tanpa harus mengubah ID kunci atau kunci rahasia. Meskipun kuncinya tidak aktif, setiap upaya untuk menggunakannya dalam permintaan ke AWS API gagal dengan akses kesalahan ditolak.
4. Pada <access key ID>kotak dialog Hapus, pilih Nonaktifkan, masukkan ID kunci akses untuk mengonfirmasi bahwa Anda ingin menghapusnya, lalu pilih Hapus.

AWS CLI & SDKs

Untuk menghapus kunci akses untuk pengguna root

> **Izin minimum**
>
> Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:
>
> - Anda harus masuk sebagai pengguna Akun AWS root, yang tidak memerlukan izin tambahan AWS Identity and Access Management (IAM). Anda tidak dapat melakukan langkah-langkah ini sebagai pengguna atau peran IAM.

- AWS CLI: aws iam delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Perintah ini tidak menghasilkan output saat berhasil.
- AWS API: DeleteAccessKey

Tugas yang memerlukan kredensial pengguna root

Kami menyarankan Anda mengonfigurasi pengguna administratif AWS IAM Identity Center untuk melakukan tugas sehari-hari dan mengakses AWS sumber daya. Namun, Anda dapat melakukan tugas yang tercantum di bawah ini hanya saat masuk sebagai pengguna root akun.

Untuk menyederhanakan pengelolaan kredensyal pengguna root istimewa di seluruh akun anggota di AWS Organizations, Anda dapat mengaktifkan akses root terpusat untuk membantu Anda mengamankan akses yang sangat istimewa secara terpusat ke akun Anda. Akun AWSMengelola akses root secara terpusat untuk akun anggotamemungkinkan Anda menghapus dan mencegah pemulihan kredensi pengguna root jangka panjang secara terpusat, meningkatkan keamanan akun di organisasi Anda. Setelah mengaktifkan fitur ini, Anda dapat melakukan tugas istimewa berikut di akun anggota.

- Hapus kredensi pengguna root akun anggota untuk mencegah pemulihan akun pengguna root. Anda juga dapat mengizinkan pemulihan kata sandi untuk memulihkan kredensi pengguna root untuk akun anggota.
- Hapus kebijakan bucket yang salah konfigurasi yang menolak semua prinsipal mengakses bucket Amazon S3.
- Hapus kebijakan berbasis sumber daya Amazon Simple Queue Service yang menolak semua prinsipal mengakses antrian Amazon SQS.

Tugas Manajemen Akun

- Ubah Akun AWS pengaturan Anda. Standalone Akun AWS yang bukan bagian dari AWS Organizations memerlukan kredensyal root untuk memperbarui alamat email, kata sandi pengguna root, dan kunci akses pengguna root. Pengaturan akun lainnya, seperti nama akun, informasi kontak, kontak alternatif, preferensi mata uang pembayaran, dan Wilayah AWS, tidak memerlukan kredensi pengguna root.

  > **Note**
  >
  > AWS Organizations, dengan semua fitur diaktifkan, dapat digunakan untuk mengelola pengaturan akun anggota secara terpusat dari akun manajemen dan akun admin yang didelegasikan. Pengguna IAM resmi atau peran IAM di akun manajemen dan akun admin yang didelegasikan dapat menutup akun anggota dan memperbarui alamat email root, nama akun, informasi kontak, kontak alternatif, dan akun Wilayah AWS anggota.

- Tutup Anda Akun AWS. Standalone Akun AWS yang bukan bagian dari AWS Organizations memerlukan kredensi root untuk menutup akun. Dengan AWS Organizations, Anda dapat menutup akun anggota secara terpusat dari akun manajemen dan akun admin yang didelegasikan.
- Buka kembali Anda Akun AWS. Jika Anda menutup akun mandiri sendiri dan tidak memiliki saldo terutang, Anda dapat membukanya kembali selama periode pasca-penutupan. Anda harus masuk sebagai pengguna root untuk melakukannya.
- Kembalikan izin pengguna IAM. Jika satu-satunya administrator IAM secara tidak sengaja mencabut izinnya sendiri, Anda dapat masuk sebagai pengguna root untuk mengedit kebijakan dan memulihkan izin tersebut.

Tugas Penagihan

- Aktifkan akses IAM ke konsol Manajemen Penagihan dan Biaya.
- Beberapa tugas Penagihan terbatas pada pengguna root. Lihat Mengelola Panduan AWS Billing Pengguna Akun AWS di untuk informasi selengkapnya.
- Lihat faktur pajak tertentu. Pengguna IAM dengan aws-portal: ViewBilling izin dapat melihat dan mengunduh faktur PPN dari AWS Eropa, tetapi tidak AWS Inc. atau (AISPL). Amazon Internet Services Private Limited

AWS GovCloud (US) Tugas

- Daftar AWS GovCloud (US).
- Minta kunci akses pengguna root AWS GovCloud (US) akun dari AWS Dukungan.

Tugas Amazon EC2

- Mendaftar sebagai penjual di Marketplace Instans Cadangan.

AWS KMS Tugas

- Jika AWS Key Management Service kunci menjadi tidak dapat dikelola, administrator dapat memulihkannya dengan menghubungi Dukungan; Namun, Dukungan menanggapi nomor telepon utama pengguna root Anda untuk otorisasi dengan mengonfirmasi OTP tiket.

Tugas Layanan Penyimpanan Sederhana Amazon

- Konfigurasikan bucket Amazon S3 untuk mengaktifkan MFA (otentikasi multi-faktor).
- Edit atau hapus kebijakan bucket Amazon S3 yang menyangkal semua prinsipal.

  Anda dapat menggunakan tindakan istimewa untuk membuka kunci bucket Amazon S3 dengan kebijakan bucket yang salah dikonfigurasi. Lihat perinciannya di Melakukan tugas istimewa pada AWS Organizations akun anggota.

Tugas Layanan Antrian Sederhana Amazon

- Mengedit atau menghapus kebijakan berbasis sumber daya Amazon SQS yang menyangkal semua prinsipal.

  Anda dapat menggunakan tindakan istimewa untuk membuka kunci antrean Amazon SQS dengan kebijakan berbasis sumber daya yang salah dikonfigurasi. Lihat perinciannya di Melakukan tugas istimewa pada AWS Organizations akun anggota.

Sumber daya tambahan

Untuk informasi selengkapnya tentang pengguna AWS root, lihat sumber daya berikut:

- Untuk bantuan terkait masalah pengguna root, lihatMemecahkan masalah dengan pengguna root.
- Untuk mengelola alamat email pengguna root secara terpusat AWS Organizations, lihat Memperbarui alamat email pengguna root untuk akun anggota di Panduan AWS Organizations Pengguna.

Artikel berikut memberikan informasi tambahan tentang bekerja dengan pengguna root.

- Apa saja praktik terbaik untuk mengamankan sumber daya saya Akun AWS dan sumber dayanya?
- Bagaimana cara membuat aturan EventBridge acara untuk memberi tahu saya bahwa pengguna root saya digunakan?
- Memantau dan memberi tahu aktivitas Pengguna root akun AWS
- Pantau aktivitas pengguna root IAM

Pengguna IAM:

> **Important**
>
> Praktik terbaik IAM menyarankan agar Anda mewajibkan pengguna manusia menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara alih-alih menggunakan pengguna IAM dengan kredenSIAL jangka panjang. Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.

> Jika Anda menggunakan AWS pengalaman baru kami dan ingin memberi pengguna manusia akses ke proyek atau AWS Pengaturan Anda, Anda harus menambahkannya ke tim Anda. Dalam pengalaman baru kami, Anda tidak dapat membuat pengguna IAM dengan akses konsol. Untuk informasi selengkapnya, lihat Mengundang anggota tim.

Pengguna IAM adalah entitas yang Anda buat di. Akun AWS Pengguna IAM mewakili pengguna manusia atau beban kerja yang menggunakan pengguna IAM untuk berinteraksi dengan AWS sumber daya. Pengguna IAM terdiri dari nama dan kredenSIAL.

Pengguna IAM dengan izin administrator tidak sama dengan. Pengguna root akun AWS Untuk informasi lebih lanjut tentang pengguna akar, lihat Pengguna root akun AWS.

Bagaimana AWS mengidentifikasi pengguna IAM

Saat Anda membuat pengguna IAM, IAM membuat cara-cara berikut untuk mengidentifikasi pengguna tersebut:

- “Nama ramah” untuk pengguna IAM, yang merupakan nama yang Anda tentukan saat membuat pengguna IAM, seperti `Richard` atau. `Anaya` Ini adalah nama yang Anda lihat di Konsol Manajemen AWS. Karena nama pengguna IAM muncul di Nama Sumber Daya Amazon (ARN), kami tidak menyarankan menyertakan informasi pengenal pribadi dalam nama IAM. Lihat persyaratan dan batasan untuk nama IAM. Persyaratan nama IAM
- Nama Sumber Daya Amazon (ARN) untuk pengguna IAM. Anda menggunakan ARN saat Anda perlu mengidentifikasi pengguna IAM secara unik di semua. AWS Misalnya, Anda dapat menggunakan ARN untuk menentukan pengguna IAM sebagai a `Principal` dalam kebijakan IAM untuk bucket Amazon S3. ARN untuk pengguna IAM mungkin terlihat seperti berikut ini:

  `arn:aws:iam::`*`account-ID-without-hyphens`*`:user/Richard`
- Pengidentifikasi unik untuk pengguna IAM. ID ini dikembalikan hanya saat Anda menggunakan API, Tools for Windows PowerShell, atau AWS CLI untuk membuat pengguna IAM; Anda tidak melihat ID ini di konsol.

Untuk informasi selengkapnya tentang pengidentifikasi ini, lihat Pengidentifikasi IAM.

Pengguna dan kredenSIAL IAM

Anda dapat mengakses dengan AWS cara yang berbeda tergantung pada kredentif pengguna IAM:

- Kata sandi konsol : Kata sandi yang dapat diketikkan pengguna IAM untuk masuk ke sesi interaktif seperti. Konsol Manajemen AWS Menonaktifkan kata sandi (akses konsol) untuk pengguna IAM mencegah mereka masuk ke Konsol Manajemen AWS menggunakan kredenSIAL masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Pengguna IAM dengan akses konsol diaktifkan juga dapat menggunakan kredenSIAL yang sama ini untuk mengotentikasi AWS CLI dan akses SDK menggunakan perintah. `aws login` AWS CLI Pengguna ini harus memiliki SignInLocalDevelopmentAccess izin. Lihat Otentikasi dan kredentif akses untuk AWS CLI di Panduan AWS Command Line Interface Pengguna untuk detail selengkapnya.
- Tombol akses : Digunakan untuk melakukan panggilan terprogram ke AWS. Namun, ada alternatif yang lebih aman untuk dipertimbangkan sebelum Anda membuat kunci akses untuk pengguna IAM. Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang di Referensi Umum AWS. Jika pengguna IAM memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell, AWS API, atau Aplikasi Seluler Kon AWS sol.
- Kunci SSH untuk digunakan dengan CodeCommit: Kunci publik SSH dalam format OpenSSH yang dapat digunakan untuk diautentikasi dengan CodeCommit.
- Sertifikat server : SSL/TLS sertifikat yang dapat Anda gunakan untuk mengotentikasi dengan beberapa AWS layanan. Kami menyarankan Anda menggunakan AWS Certificate Manager (ACM) untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Gunakan IAM hanya ketika Anda harus mendukung koneksi HTTPS di wilayah yang tidak didukung oleh ACM. Untuk mempelajari wilayah mana yang mendukung ACM, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

Anda dapat memilih kredensial yang tepat untuk pengguna IAM Anda. Saat Anda menggunakan Konsol Manajemen AWS untuk membuat pengguna IAM, Anda harus memilih untuk setidaknya menyertakan kata sandi konsol atau kunci akses. Secara default, pengguna IAM baru yang dibuat menggunakan AWS API AWS CLI atau tidak memiliki kredenSIAL dalam bentuk apa pun. Anda harus membuat jenis kredenSIAL untuk pengguna IAM berdasarkan kasus penggunaan Anda.

Anda memiliki opsi berikut untuk mengelola kata sandi, kunci akses, dan perangkat otentikasi multi-faktor (MFA):

- Kelola kata sandi untuk pengguna IAM Anda. Buat dan ubah kata sandi yang mengizinkan akses ke Konsol Manajemen AWS. Atur kebijakan kata sandi untuk menerapkan kerumitan kata sandi minimum. Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri.

- Kelola kunci akses untuk pengguna IAM Anda. Buat dan perbarui kunci akses untuk akses terprogram ke sumber daya di akun Anda.
- Aktifkan otentikasi multi-faktor (MFA) untuk pengguna IAM. Sebagai praktik terbaik, kami menyarankan Anda memerlukan otentikasi multi-faktor untuk semua pengguna IAM di akun Anda. Dengan MFA, pengguna IAM harus memberikan dua bentuk identifikasi: Pertama, mereka memberikan kredenSIAL yang merupakan bagian dari identitas pengguna mereka (kata sandi atau kunci akses). Selain itu, mereka menyediakan kode numerik sementara yang dihasilkan pada perangkat keras atau oleh aplikasi di smartphone atau tablet.
- Temukan kata sandi dan kunci akses yang tidak digunakan. Siapa pun yang memiliki kata sandi atau kunci akses untuk akun Anda atau pengguna IAM di akun Anda memiliki akses ke AWS sumber daya Anda. Praktik terbaik keamanan adalah menghapus kata sandi dan kunci akses ketika pengguna IAM tidak lagi membutuhkannya.
- Unduh laporan kredensi untuk akun Anda. Anda dapat membuat dan mengunduh laporan kredensial yang mencantumkan semua pengguna IAM di akun Anda dan status berbagai kredensial mereka, termasuk kata sandi, kunci akses, dan perangkat MFA. Untuk kata sandi dan access key, laporan kredensial menunjukkan bagaimana kata sandi atau access key telah digunakan baru-baru ini.

Pengguna dan izin IAM

Secara default, pengguna IAM baru tidak memiliki izin untuk melakukan apa pun. Mereka tidak berwenang untuk melakukan AWS operasi apa pun atau mengakses AWS sumber daya apa pun. Keuntungan memiliki pengguna IAM individual adalah Anda dapat memberikan izin secara individu kepada setiap pengguna. Anda dapat menetapkan izin administratif untuk beberapa pengguna, yang kemudian dapat mengelola AWS sumber daya Anda dan bahkan dapat membuat dan mengelola pengguna IAM lainnya. Namun, dalam kebanyakan kasus, Anda ingin membatasi izin pengguna hanya pada tugas (AWS tindakan atau operasi) dan sumber daya yang diperlukan untuk pekerjaan tersebut.

Bayangkan seorang pengguna bernama Diego. Saat Anda membuat pengguna IAM`Diego`, Anda membuat kata sandi untuknya dan melampirkan izin yang memungkinkannya meluncurkan instans Amazon EC2 tertentu dan membaca (`GET`) informasi dari tabel di database Amazon RDS. Untuk prosedur tentang cara membuat pengguna IAM dan memberi mereka kredentif dan izin awal, lihatBuat pengguna IAM di Akun AWS. Untuk prosedur tentang cara mengubah izin untuk pengguna yang sudah ada, lihat Mengubah izin untuk pengguna IAM. Untuk prosedur tentang cara mengubah

kata sandi atau access key pengguna, lihat Kata sandi pengguna di AWS dan Kelola access key untuk pengguna IAM.

Anda juga dapat menambahkan batas izin ke pengguna IAM Anda. Batas izin adalah fitur lanjutan yang memungkinkan Anda menggunakan kebijakan AWS terkelola untuk membatasi izin maksimum yang dapat diberikan kebijakan berbasis identitas kepada pengguna atau peran IAM. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Pengguna dan akun IAM

Setiap pengguna IAM dikaitkan dengan satu dan hanya satu Akun AWS. Karena pengguna IAM ditentukan di dalam Anda Akun AWS, mereka tidak perlu memiliki metode pembayaran yang tercatat. AWS Setiap AWS aktivitas yang dilakukan oleh pengguna IAM di akun Anda akan ditagih ke akun Anda.

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Pengguna IAM sebagai akun layanan

Pengguna IAM adalah sumber daya di IAM yang memiliki kredensial dan izin terkait. Pengguna IAM dapat mewakili seseorang atau aplikasi yang menggunakan kredensialnya untuk membuat AWS permintaan. Ini biasanya disebut sebagai akun layanan. Jika Anda memilih untuk menggunakan kredensial jangka panjang seorang pengguna IAM di aplikasi Anda, jangan menyematkan kunci akses secara langsung ke kode aplikasi Anda. AWS SDK dan AWS Command Line Interface memungkinkan Anda untuk menempatkan kunci akses di lokasi yang diketahui sehingga Anda tidak perlu menyimpannya dalam kode. Untuk informasi selengkapnya, lihat M engelola Kunci Akses Pengguna IAM dengan Ben ar di Referensi Umum AWS. Atau, sebagai praktik terbaik, Anda dapat menggunakan kredensial keamanan sementara (peran IAM) alih-alih kunci akses jangka panjang.

Setelah Anda membuat pengguna IAM, Anda dapat Lihat pengguna IAM di akun Anda dan Ganti nama pengguna IAM sesuai kebutuhan.

Cara pengguna IAM masuk AWS

Untuk masuk Konsol Manajemen AWS sebagai pengguna IAM, Anda harus memberikan ID akun atau alias akun selain nama pengguna dan kata sandi Anda. Saat administrator Anda membuat

pengguna IAM di konsol, seharusnya mereka mengirimkan kredensial masuk Anda, termasuk nama pengguna Anda dan URL ke halaman masuk akun Anda yang mencakup ID akun atau alias akun Anda.

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

> **Kiat**
>
> Untuk membuat bookmark untuk halaman masuk akun di browser web, Anda harus mengetikkan URL masuk secara manual untuk akun Anda di entri bookmark. Jangan gunakan fitur bookmark browser web karena pengalihan dapat mengaburkan URL masuk.

Anda juga dapat masuk di titik akhir masuk umum berikut dan mengetikkan ID akun atau alias akun Anda secara manual:

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

Untuk kenyamanan, AWS halaman masuk menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun IAM. Lain kali pengguna pergi ke halaman mana pun di Konsol Manajemen AWS, konsol menggunakan cookie untuk mengarahkan pengguna ke halaman masuk akun.

Anda hanya memiliki akses ke AWS sumber daya yang ditentukan administrator dalam kebijakan yang dilampirkan ke identitas pengguna IAM Anda. Untuk bekerja di konsol, Anda harus memiliki izin untuk melakukan tindakan yang dilakukan konsol, seperti mencantumkan dan membuat AWS sumber daya. Untuk informasi selengkapnya, lihat Manajemen akses untuk AWS sumber daya dan Contoh kebijakan berbasis identitas IAM.

> **Note**
>
> Jika organisasi Anda memiliki sistem identitas yang sudah ada, Anda mungkin ingin membuat satu sistem sekali masuk (SSO). SSO memberi pengguna akses ke Konsol Manajemen AWS akun Anda tanpa mengharuskan mereka memiliki identitas pengguna IAM. SSO juga menghilangkan kebutuhan pengguna untuk masuk ke situs organisasi Anda dan secara AWS terpisah. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Mencatat detail login di CloudTrail

Jika Anda mengaktifkan CloudTrail untuk mencatat peristiwa masuk ke log Anda, Anda harus mengetahui cara CloudTrail memilih tempat untuk mencatat peristiwa tersebut.

- Jika pengguna Anda masuk langsung ke konsol, mereka diarahkan ke titik akhir masuk global atau regional, berdasarkan apakah konsol layanan yang dipilih mendukung wilayah. Misalnya, halaman beranda konsol utama mendukung wilayah, jadi jika Anda masuk ke URL berikut ini:

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

  Anda diarahkan ke titik akhir masuk regional seperti`https://us-east-2.signin.aws.amazon.com`, menghasilkan entri CloudTrail log regional di log wilayah pengguna:

  Di sisi lain, konsol Amazon S3 tidak mendukung wilayah, jadi jika Anda masuk ke URL berikut ini

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

  AWS mengarahkan Anda ke titik akhir masuk global di`https://signin.aws.amazon.com`, menghasilkan entri CloudTrail log global.
- Anda dapat secara manual meminta titik akhir masuk regional tertentu dengan masuk ke beranda konsol utama yang diaktifkan wilayah menggunakan sintaks URL seperti berikut ini:

  ```
  https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
  ```

  AWS mengarahkan Anda ke titik akhir masuk `ap-southeast-1` regional dan menghasilkan peristiwa CloudTrail log regional.

Untuk informasi selengkapnya tentang CloudTrail dan IAM, lihat Pencat atan peristiwa IAM dengan. CloudTrail

Jika pengguna memerlukan akses terprogram untuk bekerja dengan akun Anda, Anda dapat membuat pasangan kunci akses (ID kunci akses dan kunci akses rahasia) untuk setiap pengguna. Namun, ada alternatif yang lebih aman untuk dipertimbangkan sebelum Anda membuat kunci akses untuk pengguna. Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang di Referensi Umum AWS.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang AWS masuk.

- Panduan AWS Sign-In Pengguna membantu Anda memahami berbagai cara untuk masuk ke Amazon Web Services (AWS), tergantung pada jenis pengguna Anda.
- Anda dapat masuk hingga lima identitas berbeda secara bersamaan dalam satu browser web di Konsol Manajemen AWS. Untuk detailnya, lihat Masuk ke beberapa akun di Panduan Mem Konsol Manajemen AWS ulai.
- Untuk menemukan ID akun atau alias yang diperlukan untuk masuk, lihat Melihat Anda Akun AWS ID danMenggunakan alias untuk Anda Akun AWS ID.

MFA mengaktifkan login

Pengguna yang dikonfigurasi dengan perangkat otentikasi multi-faktor (MFA) harus menggunakan perangkat MFA mereka untuk masuk ke perangkat MFA. Konsol Manajemen AWS Setelah pengguna memasukkan kredensialnya masuk, AWS periksa akun pengguna untuk melihat apakah MFA diperlukan untuk pengguna tersebut.

> **Important**
>
> Jika Anda menggunakan kunci akses dan kredenal kunci rahasia untuk Konsol Manajemen AWS akses langsung dengan panggilan AWS STS `GetFederationToken`API, MFA TIDAK akan diperlukan. Untuk informasi selengkapnya, lihat Menggunakan kunci akses dan kredentif kunci rahasia untuk akses konsol.

Topik berikut memberikan informasi tentang bagaimana pengguna menyelesaikan masuk saat MFA diperlukan.

Topik

- Beberapa perangkat MFA diaktifkan
- Kunci keamanan FIDO
- Perangkat MFA virtual
- Token TOTP perangkat keras

Beberapa perangkat MFA diaktifkan

Jika pengguna masuk Konsol Manajemen AWS sebagai pengguna Akun AWS root atau pengguna IAM dengan beberapa perangkat MFA diaktifkan untuk akun itu, mereka hanya perlu menggunakan satu perangkat MFA untuk masuk. Setelah pengguna mengautentikasi dengan kata sandi pengguna, mereka memilih jenis perangkat MFA mana yang ingin mereka gunakan untuk menyelesaikan otentikasi. Kemudian pengguna diminta untuk mengautentikasi dengan jenis perangkat yang mereka pilih.

Kunci keamanan FIDO

Jika MFA diperlukan untuk pengguna, akan muncul halaman masuk kedua. Pengguna perlu mengetuk kunci keamanan FIDO.

> **Note**
>
> Pengguna Google Chrome tidak boleh memilih salah satu opsi yang tersedia pada pop-up yang meminta Verifikasi identitas Anda dengan amazon.com. Anda hanya perlu mengetuk tombol keamanan.

Tidak seperti perangkat MFA lainnya, kunci keamanan FIDO tidak keluar dari sinkron. Administrator dapat menonaktifkan kunci keamanan FIDO jika hilang atau rusak. Untuk informasi selengkapnya, lihat Menonaktifkan perangkat MFA (konsol).

Untuk informasi tentang browser yang mendukung WebAuthn dan perangkat yang mendukung FIDO yang AWS mendukung, lihat. Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Perangkat MFA virtual

Jika MFA diperlukan untuk pengguna, akan muncul halaman masuk kedua. Di kotak kode MFA pengguna harus memasukkan kode numerik yang diberikan oleh aplikasi MFA.

Jika kode MFA sudah benar, pengguna dapat mengakses Konsol Manajemen AWS. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

Perangkat MFA virtual bisa jadi tidak sinkron. Jika pengguna tidak dapat masuk ke Konsol Manajemen AWS setelah beberapa kali mencoba, pengguna diminta untuk menyinkronkan perangkat MFA virtual. Pengguna dapat mengikuti perintah di layar untuk menyinkronkan perangkat MFA virtual.

Untuk informasi tentang cara menyinkronkan perangkat atas nama pengguna di perangkat Anda Akun AWS, lihatSinkronisasi ulang perangkat MFA virtual dan perangkat keras.

Token TOTP perangkat keras

Jika MFA diperlukan untuk pengguna, akan muncul halaman masuk kedua. Dalam kotak kode MFA, pengguna harus memasukkan kode numerik yang disediakan oleh token TOTP perangkat keras.

Jika kode MFA sudah benar, pengguna dapat mengakses Konsol Manajemen AWS. Jika kode salah, pengguna dapat mencoba kembali dengan kode lain.

Token TOTP perangkat keras bisa tidak sinkron. Jika pengguna tidak dapat masuk Konsol Manajemen AWS setelah beberapa kali mencoba, pengguna diminta untuk menyinkronkan perangkat token MFA. Pengguna dapat mengikuti perintah di layar untuk menyinkronkan perangkat token MFA. Untuk informasi tentang cara menyinkronkan perangkat atas nama pengguna di perangkat Anda Akun AWS, lihatSinkronisasi ulang perangkat MFA virtual dan perangkat keras.

Buat pengguna IAM di Akun AWS

> **Important**
>
> Praktik terbaik IAM menyarankan agar Anda mewajibkan pengguna manusia menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredentif sementara alih-alih menggunakan pengguna IAM dengan kredentif jangka panjang. Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan tertentu yang tidak didukung oleh pengguna federasi.
> Jika Anda menggunakan AWS pengalaman baru kami dan ingin memberi pengguna manusia akses ke proyek atau AWS Pengaturan Anda, Anda harus menambahkannya ke tim Anda. Dalam pengalaman baru kami, Anda tidak dapat membuat pengguna IAM dengan akses konsol. Untuk informasi selengkapnya, lihat Mengundang anggota tim.

Proses pembuatan pengguna IAM dan memungkinkan pengguna tersebut untuk melakukan tugas terdiri dari langkah-langkah berikut:

1. Buat pengguna di Konsol Manajemen AWS, AWS CLI, Alat untuk Windows PowerShell, atau menggunakan operasi AWS API. Jika Anda membuat pengguna di Konsol Manajemen AWS, maka langkah 1-4 ditangani secara otomatis, berdasarkan pilihan Anda. Jika Anda membuat

pengguna IAM secara terprogram, maka Anda harus melakukan masing-masing langkah tersebut secara individual.

2. Buat kredensial untuk pengguna, tergantung pada jenis akses yang diperlukan pengguna:
   - Aktifkan akses konsol — opsional : Jika pengguna perlu mengakses Konsol Manajemen AWS, buat kata sandi untuk pengguna. Menonaktifkan akses konsol untuk pengguna mencegah pengguna masuk Konsol Manajemen AWS menggunakan nama pengguna dan kata sandi mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan.

   > **Tip**
   >
   > Buat hanya kredenSIAL yang dibutuhkan pengguna. Misalnya, untuk pengguna yang memerlukan akses hanya melalui Konsol Manajemen AWS, jangan membuat kunci akses.

3. Berikan izin pengguna untuk melakukan tugas yang diperlukan. Sebaiknya masukkan pengguna IAM ke dalam grup dan mengelola izin melalui kebijakan yang dilampirkan ke grup tersebut. Namun, Anda juga dapat memberikan izin dengan melampirkan kebijakan izin langsung ke pengguna. Jika Anda menggunakan konsol untuk menambahkan pengguna, Anda dapat menyalin izin dari pengguna yang sudah ada ke pengguna baru.

   Anda juga dapat menambahkan batas izin untuk membatasi izin pengguna dengan menentukan kebijakan yang menentukan izin maksimum yang dapat dimiliki pengguna. Batas izin tidak memberikan izin apa pun.

   Untuk petunjuk tentang membuat kebijakan izin khusus untuk digunakan untuk memberikan izin atau menetapkan batas izin, lihatTentukan izin IAM khusus dengan kebijakan terkelola pelanggan.
4. (Opsional) Tambahkan metadata ke pengguna dengan melampirkan tag. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
5. Berikan informasi masuk yang diperlukan kepada pengguna. Ini mencakup kata sandi dan URL konsol untuk halaman masuk akun tempat pengguna memberikan kredensial tersebut. Untuk informasi selengkapnya, lihat Cara pengguna IAM masuk AWS.
6. (Opsional) Konfigurasi Multi-Factor Authentication (MFA) bagi pengguna. MFA mewajibkan pengguna untuk memberikan kode sekali pakai setiap kali dia masuk ke Konsol Manajemen AWS.
7. (Opsional) Berikan izin kepada pengguna IAM untuk mengelola kredenSIAL keamanan mereka sendiri. (Secara default, pengguna IAM tidak memiliki izin untuk mengelola kredensialnya sendiri.)

Untuk informasi selengkapnya, lihat Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri.

> **Note**
>
> Jika Anda menggunakan konsol untuk membuat pengguna dan Anda memilih Pengguna harus membuat kata sandi baru saat masuk berikutnya (disarankan), pengguna memiliki izin yang diperlukan.

Untuk informasi tentang izin yang Anda butuhkan untuk membuat pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Untuk petunjuk tentang membuat pengguna IAM untuk kasus penggunaan tertentu, lihat topik berikut:

- Buat pengguna IAM untuk akses darurat
- Membuat pengguna IAM untuk beban kerja yang tidak dapat menggunakan peran IAM

Lihat pengguna IAM

Anda dapat mencantumkan pengguna IAM di grup IAM Anda Akun AWS atau di grup IAM tertentu, dan mencantumkan semua grup IAM tempat pengguna berada. Untuk informasi tentang izin yang Anda butuhkan untuk membuat daftar pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Untuk mencantumkan semua pengguna IAM di akun Anda

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).

Konsol menampilkan pengguna IAM di file Anda Akun AWS.

AWS CLI

Jalankan perintah berikut:

- `aws iam list-users`

API

Panggil operasi berikut:

- `ListUsers`

Untuk daftar pengguna IAM dalam grup IAM

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Pada panel navigasi, pilih User groups (Grup pengguna).
4. Pilih nama grup pengguna.

Pengguna IAM yang merupakan anggota grup terdaftar di tab Pengguna.

AWS CLI

Jalankan perintah berikut:

- `aws iam get-group`

API

Panggil operasi berikut:

- `GetGroup`

Untuk mencantumkan semua grup IAM tempat pengguna berada

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM.
5. Pilih tab Grup untuk menampilkan daftar grup yang menyertakan pengguna saat ini.

AWS CLI

Jalankan perintah berikut:

- `aws iam list-groups-for-user`

API

Panggil operasi berikut:

- `ListGroupsForUser`

Langkah selanjutnya

Setelah Anda memiliki daftar pengguna IAM Anda, Anda dapat mengganti nama, menghapus, atau menonaktifkan pengguna IAM menggunakan prosedur berikut.

- Ganti nama pengguna IAM
- Menghapus atau menonaktifkan pengguna IAM

Ganti nama pengguna IAM

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola pengguna dan grup IAM. Sebaliknya, pengguna dan grup Anda dikelola di luar AWS dan dapat mengakses AWS sumber daya sebagai identitas federasi. Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses AWS layanan dengan menggunakan kredensi yang disediakan melalui sumber identitas. Identitas federasi menggunakan grup yang ditentukan oleh penyedia identitas mereka. Jika Anda menggunakan AWS IAM Identity Center, lihat Mengelola identitas di Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di Pusat Identitas IAM.

Amazon Web Services menawarkan beberapa alat untuk mengelola pengguna IAM di Anda Akun AWS. Anda dapat mencantumkan pengguna IAM di akun Anda atau di grup pengguna, atau daftar semua grup IAM yang menjadi anggota pengguna. Anda dapat mengganti nama atau mengubah jalur pengguna IAM. Jika Anda beralih menggunakan identitas federasi alih-alih pengguna IAM, Anda dapat menghapus pengguna IAM dari AWS akun Anda, atau menonaktifkan pengguna.

Untuk informasi lebih lanjut tentang menambahkan, mengubah, atau menghapus kebijakan terkelola untuk pengguna IAM, lihat Mengubah izin untuk pengguna IAM. Untuk informasi tentang mengelola kebijakan inline untuk pengguna IAM, lihat Menambahkan dan menghapus izin identitas IAM, Edit kebijakan IAM, dan Hapus kebijakan IAM. Sebagai praktik terbaik, gunakan kebijakan terkelola alih-alih kebijakan inline. AWS kebijakan terkelola memberikan izin untuk banyak kasus penggunaan umum. Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan oleh semua pelanggan. AWS Oleh karena itu, kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan terkelola pelanggan yang spesifik untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat AWS kebijakan terkelola. Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Untuk mempelajari tentang memvalidasi kebijakan IAM, lihat. Validasi kebijakan IAM

> **Tip**
>
> IAM Access Analyzer dapat menganalisis layanan dan tindakan yang digunakan peran IAM Anda, dan kemudian menghasilkan kebijakan berbutir halus yang dapat Anda gunakan. Setelah menguji setiap kebijakan yang dihasilkan, Anda dapat menerapkan kebijakan tersebut ke lingkungan produksi. Ini memastikan bahwa Anda hanya memberikan izin yang diperlukan untuk beban kerja Anda. Untuk informasi selengkapnya tentang pembuatan kebijakan, lihat pembuatan kebijakan IAM Access Analyzer.

Untuk informasi tentang mengelola kata sandi pengguna IAM, lihatKelola kata sandi untuk pengguna IAM.

Mengubah nama pengguna IAM

Untuk mengubah nama atau jalur pengguna, Anda harus menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API. Tidak ada opsi di konsol untuk mengubah nama pengguna. Untuk informasi tentang izin yang Anda butuhkan untuk mengubah nama pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Saat Anda mengubah nama atau alur pengguna, hal berikut terjadi:

- Kebijakan apa pun yang terlampir pada pengguna akan tetap melekat ke pengguna dengan nama baru.
- Pengguna tetap berada di grup IAM yang sama dengan nama baru.
- ID unik untuk pengguna tetap sama. Untuk informasi lebih lanjut tentang unik IDs, lihatPengidentifikasi unik.
- Semua kebijakan sumber daya atau peran yang mengacu pada pengguna sebagai utama (pengguna diberi akses) secara otomatis diperbarui untuk menggunakan nama atau jalur baru. Misalnya, setiap kebijakan berbasis antrean di Amazon SQS atau kebijakan berbasis sumber daya di Amazon S3 diperbarui secara otomatis untuk menggunakan nama dan jalur baru.

IAM tidak secara otomatis memperbarui kebijakan yang merujuk ke pengguna sebagai sumber daya untuk menggunakan nama atau alur baru; Anda harus melakukannya secara manual. Misalnya, bayangkan pengguna tersebut `Richard` memiliki kebijakan terlampir padanya yang memungkinkannya mengelola kredensial keamanannya. Jika administrator mengubah nama

`Richard` untuk `Rich`, administrator juga perlu memperbarui kebijakan tersebut untuk mengubah sumber daya dari:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

ke ini:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

Hal ini juga berlaku jika administrator mengubah alur; administrator perlu memperbarui kebijakan agar mencerminkan jalur baru bagi pengguna.

Untuk mengganti nama pengguna

- AWS CLI: aws iam update-user
- AWS API: UpdateUser

Menghapus atau menonaktifkan pengguna IAM

Praktik terbaik merekomendasikan agar Anda menghapus pengguna IAM yang tidak digunakan dari Anda. Akun AWS Jika Anda ingin mempertahankan kredensi pengguna IAM untuk penggunaan di masa mendatang, alih-alih menghapusnya dari akun, Anda dapat menonaktifkan akses pengguna. Untuk informasi selengkapnya, lihat Menonaktifkan pengguna IAM.

> **Warning**
>
> Setelah pengguna IAM dan kunci aksesnya dihapus, mereka tidak dapat dipulihkan atau dipulihkan.

Prasyarat — Lihat akses pengguna IAM

Sebelum Anda menghapus pengguna, tinjau aktivitas tingkat layanan terbaru mereka. Ini membantu mencegah penghapusan akses dari kepala sekolah (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus pengguna IAM (konsol)

Ketika Anda menggunakan Konsol Manajemen AWS untuk menghapus pengguna IAM, IAM secara otomatis menghapus informasi terkait berikut:

- Pengidentifikasi pengguna IAM
- Setiap keanggotaan grup — yaitu, pengguna IAM dihapus dari grup mana pun yang menjadi anggota pengguna IAM
- Kata sandi apa pun yang terkait dengan pengguna IAM
- Semua kebijakan inline yang disematkan di pengguna IAM (kebijakan yang diterapkan pada pengguna IAM menggunakan izin grup pengguna tidak terpengaruh)

> **Note**
>
> IAM menghapus kebijakan terkelola apa pun yang dilampirkan ke pengguna IAM saat Anda menghapus pengguna, tetapi tidak menghapus kebijakan terkelola.

- Semua perangkat MFA yang terkait

Untuk menghapus pengguna IAM (konsol)

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Pengguna, lalu pilih kotak centang di sebelah nama pengguna IAM yang ingin Anda hapus.
4. Pilih Hapus di bagian atas halaman.

> **Note**
>
> Jika ada pengguna yang memiliki kunci akses aktif, Anda harus menonaktifkan kunci akses sebelum menghapus pengguna. Untuk informasi selengkapnya, lihat Untuk menonaktifkan kunci akses untuk pengguna IAM.

5. Dalam kotak dialog konfirmasi, masukkan nama pengguna dalam bidang input teks untuk mengonfirmasi penghapusan pengguna. Pilih Hapus.

Konsol menampilkan pemberitahuan status bahwa pengguna IAM telah dihapus.

Menghapus pengguna IAM (AWS CLI)

Berbeda dengan Konsol Manajemen AWS, ketika Anda menghapus pengguna IAM dengan AWS CLI, Anda harus menghapus item yang dilampirkan ke pengguna IAM secara manual. Prosedur ini menggambarkan proses.

Untuk menghapus pengguna IAM dari Akun AWS ()AWS CLI

1. Hapus kata sandi pengguna, jika pengguna memilikinya.

   `aws iam delete-login-profile`
2. Hapus kunci akses pengguna, jika pengguna memilikinya.

   `aws iam list-access-keys` (untuk mencantumkan kunci akses pengguna) dan `aws iam delete-access-key`
3. Hapus sertifikat masuk pengguna. Perhatikan bahwa ketika Anda menghapus kredensial keamanan, akan hilang selamanya dan tidak dapat dipulihkan.

   `aws iam list-signing-certificates` (untuk mencantumkan sertifikat masuk pengguna) dan `aws iam delete-signing-certificate`
4. Hapus kunci publik SSH pengguna, jika pengguna memilikinya.

   `aws iam list-ssh-public-keys` (untuk mencantumkan kunci publik SSH pengguna) dan `aws iam delete-ssh-public-key`
5. Hapus kredensial Git pengguna.

   `aws iam list-service-specific-credentials` (untuk mencantumkan kredensial git pengguna) dan `aws iam delete-service-specific-credential`
6. Nonaktifkan perangkat Multi-Factor Authentication (MFA) pengguna, jika pengguna memilikinya.

   `aws iam list-mfa-devices` (untuk mencantumkan perangkat MFA pengguna), `aws iam deactivate-mfa-device` (untuk menonaktifkan perangkat), dan `aws iam delete-virtual-mfa-device` (untuk menghapus perangkat MFA virtual secara permanen)

7. Hapus kebijakan inline pengguna.

   `aws iam list-user-policies` (untuk mencantumkan kebijakan inline bagi pengguna) dan `aws iam delete-user-policy` (untuk menghapus kebijakan)

8. Lepaskan kebijakan terkelola yang melekat pada pengguna.

   `aws iam list-attached-user-policies` (untuk mencantumkan kebijakan terkelola yang melekat pada pengguna) dan `aws iam detach-user-policy` (untuk melepaskan kebijakan)

9. Hapus pengguna dari grup IAM apa pun.

   `aws iam list-groups-for-user`(untuk mencantumkan grup IAM yang menjadi milik pengguna) dan `aws iam remove-user-from-group`

10. Hapus pengguna.

    `aws iam delete-user`

Menonaktifkan pengguna IAM

Anda mungkin perlu menonaktifkan pengguna IAM saat mereka sementara jauh dari perusahaan Anda. Anda dapat meninggalkan kredensi pengguna IAM mereka di tempat dan masih memblokir akses mereka. AWS

Untuk menonaktifkan pengguna, buat dan lampirkan kebijakan untuk menolak akses pengguna. AWS Anda dapat memulihkan akses pengguna nanti.

Berikut adalah dua contoh kebijakan penolakan yang dapat Anda lampirkan ke pengguna untuk menolak akses mereka.

Kebijakan berikut tidak termasuk batas waktu. Anda harus menghapus kebijakan untuk memulihkan akses pengguna.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
```

```
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

Kebijakan berikut mencakup kondisi yang memulai kebijakan pada 24 Desember 2024 pukul 23.59 (UTC) dan berakhir pada 28 Februari 2025 pukul 23.59 (UTC).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Kontrol akses pengguna IAM ke Konsol Manajemen AWS

Pengguna IAM dengan izin yang masuk ke Anda Akun AWS melalui Konsol Manajemen AWS dapat mengakses AWS sumber daya Anda. Daftar berikut menunjukkan cara-cara yang dapat Anda berikan kepada pengguna IAM akses ke Akun AWS sumber daya Anda melalui. Konsol Manajemen AWS Ini juga menunjukkan bagaimana pengguna IAM dapat mengakses fitur AWS akun lain melalui AWS situs web.

> **Note**
>
> Tidak ada biaya untuk menggunakan IAM.

The Konsol Manajemen AWS

Anda membuat kata sandi untuk setiap pengguna IAM yang membutuhkan akses ke file. Konsol Manajemen AWS Pengguna mengakses konsol melalui halaman masuk berkemampuan IAM Akun AWS . Untuk informasi tentang mengakses halaman login, lihat Cara masuk di AWS Sign-In Panduan Pengguna. AWS Untuk informasi tentang cara membuat kata sandi, lihat Kata sandi pengguna di AWS.

Anda dapat mencegah pengguna IAM mengakses Konsol Manajemen AWS dengan menghapus kata sandi mereka. Hal ini mencegah mereka masuk ke dalam Konsol Manajemen AWS menggunakan kredensi masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows, AWS API PowerShell, atau AWS Console Mobile Application.

AWS Sumber daya Anda, seperti instans Amazon EC2, bucket Amazon S3, dan sebagainya

Bahkan jika pengguna IAM Anda memiliki kata sandi, mereka masih memerlukan izin untuk mengakses AWS sumber daya Anda. Saat Anda membuat pengguna IAM, pengguna tersebut tidak memiliki izin secara default. Untuk memberi pengguna IAM izin yang mereka butuhkan, Anda melampirkan kebijakan kepada mereka. Jika Anda memiliki banyak pengguna IAM yang melakukan tugas yang sama dengan sumber daya yang sama, Anda dapat menetapkan pengguna IAM tersebut ke grup. Kemudian tetapkan izin untuk grup tersebut. Untuk informasi tentang membuat pengguna dan grup IAM, lihatIdentitas IAM. Untuk informasi tentang menggunakan kebijakan untuk mengatur izin, lihat Manajemen akses untuk AWS sumber daya.

AWS Forum Diskusi

Siapa pun dapat membaca tulisan di Forum Diskusi AWS. Pengguna yang ingin memposting pertanyaan atau komentar ke Forum AWS Diskusi dapat melakukannya menggunakan nama pengguna mereka. Saat pertama kali pengguna memposting ke Forum AWS Diskusi, pengguna diminta untuk memasukkan nama panggilan dan alamat email. Hanya pengguna yang dapat menggunakan nama panggilan itu di Forum AWS Diskusi.

Informasi Akun AWS penagihan dan penggunaan Anda

Anda dapat memberi pengguna akses informasi Akun AWS penagihan dan penggunaan Anda. Untuk informasi selengkapnya, lihat Mengontrol Akses ke Informasi Penagihan Anda di Panduan AWS Billing Pengguna.

Informasi Akun AWS profil Anda

Pengguna tidak dapat mengakses informasi Akun AWS profil Anda.

Kredensi Akun AWS keamanan Anda

Pengguna tidak dapat mengakses Akun AWS kredensil keamanan Anda.

> **Note**
>
> Kebijakan IAM mengontrol akses terlepas dari antarmuka. Misalnya, Anda dapat memberikan pengguna dengan kata sandi untuk mengakses Konsol Manajemen AWS. Kebijakan untuk pengguna tersebut (atau grup apa pun yang dimiliki pengguna) akan mengontrol apa yang dapat dilakukan pengguna di Konsol Manajemen AWS. Atau, Anda bisa memberi pengguna kunci AWS akses untuk melakukan panggilan API AWS. Kebijakan ini akan mengontrol tindakan mana yang dapat pengguna hubungi melalui perpustakaan atau klien yang menggunakan access key tersebut untuk autentikasi.

Mengubah izin untuk pengguna IAM

Anda dapat mengubah izin untuk pengguna IAM di Anda Akun AWS dengan mengubah keanggotaan grupnya, dengan menyalin izin dari pengguna yang ada, dengan melampirkan kebijakan langsung ke pengguna, atau dengan menetapkan batas izin. Batas izin mengontrol izin maksimum yang dapat dimiliki pengguna. Batas izin adalah AWS fitur lanjutan.

Untuk informasi tentang izin yang Anda butuhkan untuk memodifikasi izin pengguna, lihat Izin diperlukan untuk mengakses sumber daya IAM.

Topik

- Lihat akses pengguna
- Menghasilkan kebijakan berdasarkan aktivitas akses pengguna
- Menambahkan izin ke pengguna (konsol)
- Mengubah izin untuk pengguna (konsol)
- Untuk menghapus kebijakan izin dari pengguna (konsol)
- Untuk menghapus batas izin dari pengguna (konsol)

- Menambahkan dan menghapus izin pengguna (AWS CLI atau AWS API)

Lihat akses pengguna

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghasilkan kebijakan berdasarkan aktivitas akses pengguna

Terkadang Anda dapat memberikan izin ke entitas IAM (pengguna atau peran) di luar yang mereka butuhkan. Untuk membantu Anda menyaring izin yang Anda berikan, Anda dapat membuat kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas. IAM Access Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam rentang tanggal yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin berbutir halus, lalu melampirkannya ke entitas IAM. Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat Pembuatan kebijakan IAM Access Analyzer.

Menambahkan izin ke pengguna (konsol)

IAM menawarkan tiga cara untuk menambahkan kebijakan izin kepada pengguna:

- Tambahkan pengguna IAM ke grup IAM — Jadikan pengguna sebagai anggota grup. Kebijakan dari grup melekat ke pengguna.
- Salin izin dari pengguna IAM yang ada — Salin semua keanggotaan grup, kebijakan terkelola terlampir, kebijakan sebaris, dan batasan izin yang ada dari pengguna sumber.
- Lampirkan kebijakan langsung ke pengguna IAM — Lampirkan kebijakan terkelola langsung ke pengguna. Untuk pengelolaan izin yang lebih mudah, lampirkan kebijakan Anda ke grup, lalu jadikan anggota pengguna IAM dari grup yang sesuai.

> **Important**
>
> Jika pengguna memiliki batas izin, maka Anda tidak dapat menambahkan lebih banyak izin kepada pengguna daripada yang diizinkan oleh batas izin.

Untuk menambahkan izin dengan menambahkan pengguna IAM ke grup

Menambahkan pengguna IAM ke grup IAM akan segera memperbarui izin pengguna dengan izin yang ditentukan untuk grup tersebut.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM.
5. Pilih tab Grup untuk menampilkan daftar grup yang menyertakan pengguna saat ini.
6. Pilih Tambahkan pengguna ke grup.
7. Pilih kotak centang untuk setiap grup yang Anda ingin pengguna bergabung. Daftar ini menunjukkan nama masing-masing grup dan kebijakan yang diterima pengguna jika telah menjadi anggota grup tersebut.
8. (Opsional) Anda dapat memilih Buat grup untuk menentukan grup baru. Ini berguna jika Anda ingin menambahkan pengguna ke grup dengan kebijakan terlampir yang berbeda dari grup yang ada:

   a. Di tab baru, untuk Nama grup pengguna, ketikkan nama untuk grup baru Anda.

   > **Note**
   >
   > Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota. Nama grup dapat berupa kombinasi hingga 128 huruf, digit, dan karakter ini: plus (+), sama dengan (=), koma (,), titik (.), tanda a keong (@), dan tanda hubung (-). Nama harus unik dalam akun. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat dua grup yang diberi nama TESTGROUP dan testgroup.

   b. Pilih satu atau beberapa kotak centang untuk kebijakan terkelola yang ingin dilampirkan ke grup. Anda juga dapat membuat kebijakan pengelolaan baru dengan memilih Buat kebijakan. Jika Anda melakukannya, kembali ke tab atau jendela browser ini ketika

kebijakan baru selesai; pilih Segarkan; lalu pilih kebijakan baru untuk dilekatkan ke grup Anda. Untuk informasi selengkapnya, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.

c. Pilih Buat grup pengguna.

d. Kembali ke tab asal, muat ulang daftar grup Anda. Kemudian pilih kotak centang untuk grup baru Anda.

9. Pilih Tambahkan pengguna ke grup.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa pengguna telah ditambahkan ke grup yang Anda tentukan.

Untuk menambahkan izin dengan menyalin dari pengguna IAM lain

Jika Anda memilih untuk menambahkan izin ke pengguna IAM dengan menyalin izin, IAM akan menyalin semua keanggotaan grup, kebijakan terkelola terlampir, kebijakan sebaris, dan batasan izin yang ada dari pengguna yang ditentukan dan langsung menerapkannya ke pengguna yang dipilih saat ini.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM.
5. Pada tab Izin pilih dan pilih Tambahkan izin.
6. Pada halaman Tambahkan izin, pilih Salin izin. Daftar ini menampilkan pengguna IAM yang tersedia bersama dengan keanggotaan grup dan kebijakan terlampir mereka.
7. Pilih tombol radio di samping pengguna yang izinnya ingin Anda salin.
8. Pilih Berikutnya untuk melihat daftar perubahan yang akan dilakukan pada pengguna. Lalu, pilih Tambahkan izin.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa izin telah disalin dari pengguna IAM yang Anda tentukan.

Untuk menambahkan izin dengan melampirkan kebijakan langsung ke pengguna IAM

Anda dapat melampirkan kebijakan terkelola langsung ke pengguna IAM. Izin yang diperbarui diterapkan segera.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM.
5. Pada tab Izin, pilih dan pilih Tambahkan izin.
6. Pada halaman Tambahkan izin, pilih Lampirkan kebijakan secara langsung. Daftar kebijakan izin menampilkan kebijakan yang tersedia beserta jenis kebijakan dan entitas terlampirnya.
7. Pilih tombol radio di sebelah Nama kebijakan yang ingin Anda lampirkan.
8. Pilih Berikutnya untuk melihat daftar perubahan yang akan dilakukan pada pengguna. Lalu, pilih Tambahkan izin.

   Konsol menampilkan pesan status yang memberi tahu Anda bahwa kebijakan telah ditambahkan ke pengguna IAM yang Anda tentukan.

Untuk mengatur batas izin untuk pengguna IAM

Batas izin adalah fitur lanjutan untuk mengelola izin yang digunakan untuk mengatur izin maksimum AWS yang dapat dimiliki pengguna IAM. Menetapkan batas izin segera membatasi izin pengguna IAM ke batas, terlepas dari izin lain yang diberikan.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).

4. Dalam daftar Pengguna, pilih nama pengguna IAM yang batas izinnya ingin Anda ubah.
5. Pilih tab Izin. Jika perlu, buka bagian batas Izin dan kemudian pilih Tetapkan batas izin.
6. Pada halaman Setel batas izin, di bawah Kebijakan izin pilih kebijakan yang ingin Anda gunakan untuk batas izin.
7. Pilih Tetapkan batas.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa batas izin telah ditambahkan.

Mengubah izin untuk pengguna (konsol)

IAM memungkinkan Anda mengubah izin yang terkait dengan pengguna dalam cara berikut:

- Edit kebijakan izin – Edit kebijakan inline pengguna, kebijakan inline grup pengguna, atau edit kebijakan yang terkelola yang dilampirkan pada pengguna secara langsung atau dari grup. Jika pengguna memiliki batas izin, maka Anda tidak dapat memberikan izin lebih dari yang diizinkan oleh kebijakan yang digunakan sebagai batas izin pengguna.
- Mengubah batas izin – Ubah kebijakan yang digunakan sebagai batas izin bagi pengguna. Ini dapat memperluas atau membatasi izin maksimum yang dapat dimiliki pengguna.

Mengedit kebijakan izin yang terlampir pada pengguna

Mengubah izin segera memperbarui akses pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM yang batas izinnya ingin Anda ubah.
5. Pilih tab Izin. Jika perlu, buka bagian batas Izin.
6. Pilih nama kebijakan yang ingin Anda edit untuk melihat perincian tentang kebijakan tersebut. Pilih tab Entitas yang dilampirkan untuk melihat entitas lain (pengguna, grup, dan peran IAM) yang mungkin terpengaruh jika Anda mengedit kebijakan.

7. Pilih Tab Izin dan meninjau izin yang diberikan oleh kebijakan. Untuk membuat perubahan pada izin, pilih Edit.
8. Edit kebijakan dan selesaikan rekomendasi validasi kebijakan apa pun. Untuk informasi selengkapnya, lihat Edit kebijakan IAM.
9. Pilih Berikutnya, tinjau ringkasan kebijakan, lalu pilih Simpan perubahan.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa kebijakan telah diperbarui.

Untuk mengubah batas izin bagi pengguna

Mengubah batas izin akan segera memperbarui akses pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM yang batas izinnya ingin Anda ubah.
5. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Ubah batas.
6. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
7. Pilih Tetapkan batas.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa batas izin telah diubah.

Untuk menghapus kebijakan izin dari pengguna (konsol)

Menghapus kebijakan izin akan segera memperbarui akses pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.

2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Users (Pengguna).
4. Pilih nama pengguna yang kebijakan izinnya ingin Anda hapus.
5. Pilih tab Izin.
6. Jika Anda ingin menghapus izin dengan menghapus kebijakan yang ada, lihat kolom Terlampir melalui untuk memahami cara pengguna mendapatkan kebijakan tersebut sebelum memilih Hapus untuk menghapus kebijakan:

   - Jika kebijakan berlaku karena keanggotaan grup, maka memilih Hapus akan menghapus pengguna dari grup. Ingatlah bahwa Anda mungkin memiliki beberapa kebijakan yang terlampir pada satu grup. Jika Anda menghapus pengguna dari grup, pengguna kehilangan akses ke semua kebijakan yang diterima melalui keanggotaan grup tersebut.
   - Jika kebijakan tersebut merupakan kebijakan terkelola yang dilampirkan langsung ke pengguna, maka memilih Hapus akan melepaskan kebijakan dari pengguna. Hal ini tidak memengaruhi kebijakan itu sendiri atau entitas lainnya yang mungkin menyertai kebijakan tersebut.
   - Jika kebijakan tersebut merupakan kebijakan yang disematkan sebaris, maka memilih Hapus akan menghapus kebijakan dari IAM. Kebijakan inline yang dilampirkan langsung ke pengguna hanya ada pada pengguna tersebut.

Jika kebijakan diberikan kepada pengguna melalui keanggotaan grup, konsol akan menampilkan pesan status yang memberi tahu Anda bahwa pengguna IAM telah dihapus dari grup IAM. Jika kebijakan langsung dilampirkan atau sebaris, pesan status akan memberi tahu Anda bahwa kebijakan tersebut telah dihapus.

Untuk menghapus batas izin dari pengguna (konsol)

Menghapus batas izin akan segera memperbarui akses pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.

3. Di panel navigasi, pilih Users (Pengguna).
4. Dalam daftar Pengguna, pilih nama pengguna IAM yang batas izinnya ingin Anda hapus.
5. Pilih tab Izin. Jika perlu, buka bagian batas Izin.
6. Pilih Ubah batas. Untuk mengonfirmasi bahwa Anda ingin menghapus batas izin, dalam dialog konfirmasi, pilih Hapus batas.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa batas izin telah dihapus.

Menambahkan dan menghapus izin pengguna (AWS CLI atau AWS API)

Untuk menambah atau menghapus izin secara terprogram, Anda harus menambahkan atau menghapus keanggotaan grup, melampirkan atau melepaskan kebijakan terkelola, atau menambahkan atau menghapus kebijakan inline. Untuk informasi selengkapnya, lihat topik berikut:

- Mengedit pengguna dalam grup IAM
- Menambahkan dan menghapus izin identitas IAM

Kata sandi pengguna di AWS

Anda dapat mengelola kata sandi untuk pengguna IAM di akun Anda. Pengguna IAM membutuhkan kata sandi untuk mengakses file. Konsol Manajemen AWS Pengguna tidak memerlukan kata sandi untuk mengakses AWS sumber daya secara terprogram dengan menggunakan AWS CLI, Alat untuk Windows PowerShell, atau. AWS SDKs APIs Untuk lingkungan tersebut, Anda memiliki opsi untuk menetapkan kunci akses pengguna IAM. Namun, ada alternatif lain yang lebih aman untuk mengakses kunci yang kami sarankan Anda pertimbangkan terlebih dahulu. Untuk informasi selengkapnya, lihat AWS kredensi keamanan.

> **Note**
>
> Jika salah satu pengguna IAM Anda kehilangan atau lupa kata sandi mereka, Anda tidak dapat mengambilnya dari IAM. Tergantung pada pengaturan Anda, baik pengguna atau administrator harus membuat kata sandi baru.

Konten

- Menetapkan kebijakan kata sandi akun untuk pengguna IAM

- Kelola kata sandi untuk pengguna IAM
- Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri
- Bagaimana pengguna IAM mengubah kata sandi mereka sendiri

Menetapkan kebijakan kata sandi akun untuk pengguna IAM

Anda dapat menetapkan kebijakan kata sandi khusus Akun AWS untuk menentukan persyaratan kompleksitas dan periode rotasi wajib untuk kata sandi pengguna IAM Anda. Jika Anda tidak menetapkan kebijakan kata sandi khusus, kata sandi pengguna IAM harus memenuhi kebijakan AWS kata sandi default. Untuk informasi selengkapnya, lihat Opsi kebijakan kata sandi kustom.

Topik

- Aturan untuk menetapkan kebijakan kata sandi
- Izin yang diperlukan untuk menetapkan kebijakan kata sandi
- Kebijakan kata sandi bawaan
- Opsi kebijakan kata sandi kustom
- Untuk menyetel kebijakan kata sandi (konsol)
- Untuk mengubah kebijakan kata sandi (konsol)
- Untuk menghapus kebijakan kata sandi khusus (konsol)
- Mengatur kebijakan kata sandi (AWS CLI)
- Menyetel kebijakan kata sandi (AWS API)

Aturan untuk menetapkan kebijakan kata sandi

Kebijakan kata sandi IAM tidak berlaku untuk Pengguna root akun AWS kata sandi atau kunci akses pengguna IAM. Jika kata sandi kedaluwarsa, pengguna IAM tidak dapat masuk Konsol Manajemen AWS tetapi dapat terus menggunakan kunci aksesnya.

Saat Anda membuat atau mengubah kebijakan kata sandi, sebagian besar pengaturan kebijakan kata sandi diterapkan saat pengguna Anda mengubah kata sandinya. Namun, beberapa pengaturan diterapkan dengan segera. Sebagai contoh:

- Saat persyaratan panjang dan jenis karakter minimum berubah, pengaturan ini diterapkan di lain waktu saat pengguna Anda mengubah kata sandinya. Pengguna tidak dipaksa untuk mengubah kata sandi yang sudah ada, sekalipun kata sandi yang sudah ada tidak mematuhi kebijakan kata sandi yang diperbarui.

- Saat Anda menetapkan periode kedaluwarsa kata sandi, periode kedaluwarsa tersebut akan segera diberlakukan. Misalnya, anggaplah Anda mengatur masa kedaluwarsa kata sandi selama 90 hari. Dalam hal ini, kata sandi kedaluwarsa untuk semua pengguna IAM yang kata sandinya sudah lebih lama dari 90 hari. Pengguna tersebut harus mengubah kata sandi saat masuk lagi.

Anda tidak dapat membuat “kebijakan penguncian” untuk mengunci pengguna keluar dari akun setelah sejumlah percobaan masuk gagal yang ditentukan. Untuk keamanan yang lebih baik, kami sarankan Anda menggabungkan kebijakan kata sandi yang kuat dengan multi-factor authentication (MFA). Untuk informasi lebih lanjut tentang MFA, lihat AWS Multi-factor otentikasi di IAM.

Izin yang diperlukan untuk menetapkan kebijakan kata sandi

Anda harus mengonfigurasikan izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk melihat atau mengedit kebijakan kata sandi akun mereka. Anda dapat menyertakan tindakan kebijakan kata sandi berikut dalam kebijakan IAM:

- `iam:GetAccountPasswordPolicy` – Memungkinkan entitas untuk melihat kebijakan kata sandi untuk akun mereka
- `iam:DeleteAccountPasswordPolicy` – Memungkinkan entitas untuk menghapus kebijakan kata sandi kustom untuk akun mereka dan kembali ke kebijakan kata sandi default
- `iam:UpdateAccountPasswordPolicy` – Memungkinkan entitas untuk membuat atau mengubah kebijakan kata sandi untuk akun mereka

Kebijakan berikut memungkinkan akses penuh untuk melihat dan mengedit kebijakan kata sandi akun. Untuk mempelajari cara membuat kebijakan IAM menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
```

```
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk informasi tentang izin yang diperlukan bagi pengguna IAM untuk mengubah kata sandi mereka sendiri, lihat Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri.

Kebijakan kata sandi bawaan

Jika administrator tidak menetapkan kebijakan kata sandi khusus, kata sandi pengguna IAM harus memenuhi kebijakan AWS kata sandi default.

Kebijakan kata sandi default memberlakukan kondisi berikut:

- Panjang kata sandi minimum adalah 8 karakter dan panjang maksimal 128 karakter
- Minimal tiga dari campuran tipe karakter berikut: huruf besar, huruf kecil, angka, dan karakter non-alfanumerik () ! @ # $ % ^ & * ( ) _ + - = [ ] { } | '
- Tidak identik dengan Akun AWS nama atau alamat email Anda
- Jangan pernah kedaluwarsa kata sandi

Opsi kebijakan kata sandi kustom

Saat Anda mengonfigurasi kebijakan kata sandi khusus untuk akun Anda, Anda dapat menentukan kondisi berikut:

- Panjang minimum kata sandi – Anda dapat menentukan minimal 6 karakter dan maksimal 128 karakter.
- Kekuatan kata sandi - Anda dapat memilih salah satu kotak centang berikut untuk menentukan kekuatan kata sandi pengguna IAM Anda:
  - Memerlukan setidaknya satu huruf besar dari alfabet Latin (A—Z)
  - Memerlukan setidaknya satu huruf kecil dari alfabet Latin (a-z)
  - Diperlukan setidaknya satu angka
  - Diperlukan setidaknya satu karakter nonalfanumerik ! @ # $ % ^ & * ( ) _ + - = [ ] { } | '

- Aktifkan kedaluwarsa kata sandi — Anda dapat memilih dan menentukan minimal 1 dan maksimum 1.095 hari bahwa kata sandi pengguna IAM valid setelah disetel. Misalnya, jika Anda menentukan kedaluwarsa 90 hari, itu langsung berdampak pada semua pengguna Anda. Untuk pengguna dengan kata sandi yang lebih tua dari 90 hari, ketika mereka masuk ke konsol setelah perubahan, mereka harus menetapkan kata sandi baru. Pengguna dengan kata sandi berusia 75-89 hari menerima Konsol Manajemen AWS peringatan tentang kedaluwarsa kata sandi mereka. Pengguna IAM dapat mengubah kata sandi mereka kapan saja jika mereka memiliki izin. Saat mereka mengatur kata sandi baru, periode kedaluwarsa kata sandi tersebut dimulai ulang. Pengguna IAM hanya dapat memiliki satu kata sandi yang valid dalam satu waktu.
- Kedaluwarsa kata sandi memerlukan pengaturan ulang administrator - Pilih opsi ini untuk mencegah pengguna IAM menggunakan Konsol Manajemen AWS untuk memperbarui kata sandi mereka sendiri setelah kata sandi kedaluwarsa. Sebelum Anda memilih opsi ini, konfirmasikan bahwa Anda Akun AWS memiliki lebih dari satu pengguna dengan izin administratif untuk mengatur ulang kata sandi pengguna IAM. Administrator dengan `iam:UpdateLoginProfile` izin dapat mengatur ulang kata sandi pengguna IAM. Pengguna IAM dengan `iam:ChangePassword` izin dan kunci akses aktif dapat mengatur ulang kata sandi konsol pengguna IAM mereka sendiri secara terprogram. Jika Anda menghapus kotak centang ini, pengguna IAM dengan kata sandi kedaluwarsa masih harus menetapkan kata sandi baru sebelum mereka dapat mengakses. Konsol Manajemen AWS
- Izinkan pengguna mengubah kata sandi mereka sendiri - Anda dapat mengizinkan semua pengguna IAM di akun Anda untuk mengubah kata sandi mereka sendiri. Ini memberi pengguna akses ke `iam:ChangePassword` tindakan hanya untuk pengguna mereka dan `iam:GetAccountPasswordPolicy` tindakan. Opsi ini tidak melampirkan kebijakan izin untuk setiap pengguna. Sebaliknya, IAM menerapkan izin di tingkat akun untuk semua pengguna. Atau, Anda hanya dapat mengizinkan beberapa pengguna untuk mengelola kata sandi mereka sendiri. Untuk melakukannya, Anda menghapus kotak centang ini. Untuk informasi selengkapnya tentang penggunaan kebijakan untuk membatasi siapa yang dapat mengelola kata sandi, lihat Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri.
- Cegah penggunaan ulang kata sandi – Anda dapat mencegah pengguna IAM menggunakan kembali jumlah tertentu dari kata sandi sebelumnya. Anda dapat menentukan jumlah minimum 1 dan jumlah maksimum 24 dari kata sandi sebelumnya yang dapat diulang.

Untuk menyetel kebijakan kata sandi (konsol)

Anda dapat menggunakan kebijakan Konsol Manajemen AWS untuk membuat, mengubah, atau menghapus kata sandi khusus. Perubahan pada kebijakan kata sandi berlaku untuk pengguna

IAM baru yang dibuat setelah perubahan kebijakan ini dan pengguna IAM yang ada saat mereka mengubah kata sandi mereka.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Pengaturan akun.
4. Di bagian Kebijakan kata sandi, pilih Edit.
5. Pilih Kustom untuk menggunakan kebijakan kata sandi kustom.
6. Pilih opsi yang ingin Anda terapkan ke kebijakan kata sandi Anda dan pilih Simpan perubahan.
7. Konfirmasikan bahwa Anda ingin menetapkan kebijakan kata sandi khusus dengan memilih Setel kustom.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa persyaratan kata sandi untuk pengguna IAM telah diperbarui.

Untuk mengubah kebijakan kata sandi (konsol)

Anda dapat menggunakan kebijakan Konsol Manajemen AWS untuk membuat, mengubah, atau menghapus kata sandi khusus. Perubahan pada kebijakan kata sandi berlaku untuk pengguna IAM baru yang dibuat setelah perubahan kebijakan ini dan pengguna IAM yang ada saat mereka mengubah kata sandi mereka.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Pengaturan akun.
4. Di bagian Kebijakan kata sandi, pilih Edit.

5. Pilih opsi yang ingin Anda terapkan ke kebijakan kata sandi Anda dan pilih Simpan perubahan.
6. Konfirmasikan bahwa Anda ingin menetapkan kebijakan kata sandi khusus dengan memilih Setel kustom.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa persyaratan kata sandi untuk pengguna IAM telah diperbarui.

Untuk menghapus kebijakan kata sandi khusus (konsol)

Anda dapat menggunakan kebijakan Konsol Manajemen AWS untuk membuat, mengubah, atau menghapus kata sandi khusus. Perubahan pada kebijakan kata sandi berlaku untuk pengguna IAM baru yang dibuat setelah perubahan kebijakan ini dan pengguna IAM yang ada saat mereka mengubah kata sandi mereka.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan Pengguna AWS Masuk.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Search IAM.
3. Di panel navigasi, pilih Pengaturan akun.
4. Di bagian Kebijakan kata sandi, pilih Edit.
5. Pilih IAM default untuk menghapus kebijakan kata sandi kustom dan pilih Simpan perubahan.
6. Konfirmasikan bahwa Anda ingin mengatur kebijakan kata sandi default IAM dengan memilih Set default.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa kebijakan kata sandi disetel ke default IAM.

Mengatur kebijakan kata sandi (AWS CLI)

Anda dapat menggunakan AWS Command Line Interface untuk menetapkan kebijakan kata sandi.

Untuk mengelola kebijakan kata sandi akun kustom dari AWS CLI

Jalankan perintah berikut.

- Untuk membuat atau mengubah kebijakan kata sandi kustom: `aws iam update-account-password-policy`
- Untuk melihat kebijakan kata sandi: `aws iam get-account-password-policy`
- Untuk menghapus kebijakan kata sandi khusus: `aws iam delete-account-password-policy`

Menyetel kebijakan kata sandi (AWS API)

Anda dapat menggunakan operasi AWS API untuk menetapkan kebijakan kata sandi.

Untuk mengelola kebijakan kata sandi akun kustom dari AWS API

Hubungi operasi berikut ini:

- Untuk membuat atau mengubah kebijakan kata sandi kustom: `UpdateAccountPasswordPolicy`
- Untuk melihat kebijakan kata sandi: `GetAccountPasswordPolicy`
- Untuk menghapus kebijakan kata sandi khusus: `DeleteAccountPasswordPolicy`

Kelola kata sandi untuk pengguna IAM

Pengguna IAM yang menggunakan Konsol Manajemen AWS untuk bekerja dengan AWS sumber daya harus memiliki kata sandi untuk masuk. Anda dapat membuat, mengubah, atau menghapus kata sandi untuk pengguna IAM di AWS akun Anda.

Setelah Anda menetapkan kata sandi kepada pengguna, pengguna dapat masuk Konsol Manajemen AWS menggunakan URL masuk untuk akun Anda, yang terlihat seperti ini:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Untuk informasi selengkapnya tentang cara pengguna IAM masuk ke Konsol Manajemen AWS, lihat Cara masuk AWS di Panduan AWS Sign-In Pengguna.

Sekalipun pengguna Anda memiliki kata sandi, mereka masih memerlukan izin untuk mengakses sumber daya AWS Anda. Secara default, pengguna tidak memiliki izin. Untuk memberi pengguna Anda izin yang mereka butuhkan, Anda menetapkan kebijakan untuk mereka atau ke grup yang

mereka miliki. Untuk informasi tentang membuat pengguna dan grup, lihat Identitas IAM. Untuk informasi tentang menggunakan kebijakan untuk mengatur izin, lihat Mengubah izin untuk pengguna IAM.

Anda dapat memberikan izin kepada pengguna untuk mengubah kata sandi mereka sendiri. Untuk informasi selengkapnya, lihat Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri. Untuk informasi tentang cara pengguna mengakses halaman masuk akun Anda, lihat Cara masuk AWS di Panduan AWS Sign-In Pengguna.

Topik

- Membuat, mengubah, atau menghapus kata sandi pengguna IAM (konsol)

Membuat, mengubah, atau menghapus kata sandi pengguna IAM (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk mengelola kata sandi untuk pengguna IAM Anda.

Kebutuhan akses pengguna Anda dapat berubah seiring waktu. Anda mungkin perlu mengaktifkan pengguna yang ditujukan untuk akses CLI untuk memiliki akses konsol, mengubah kata sandi pengguna karena mereka menerima email dengan kredensialnya, atau menghapus pengguna saat mereka meninggalkan organisasi Anda atau tidak lagi memerlukan AWS akses.

Untuk membuat kata sandi pengguna IAM (konsol)

Gunakan prosedur ini untuk memberikan akses konsol pengguna dengan membuat kata sandi yang terkait dengan nama pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Cari IAM.
3. Di panel navigasi, pilih Pengguna.
4. Pilih nama pengguna yang kata sandinya ingin Anda buat.
5. Pilih tab Kredensibilitas keamanan, lalu di bawah Mas uk ke konsol, pilih Aktifkan akses konsol.

6. Di kotak dialog Aktifkan akses konsol, pilih Setel ulang kata sandi, lalu pilih apakah IAM akan membuat kata sandi atau membuat kata sandi khusus:

   - Untuk membuat IAM membuat kata sandi, pilih Kata sandi yang dibuat otomatis.
   - Untuk membuat kata sandi kustom, pilih Kata sandi kustom, dan ketik kata sandi.

   > **Note**
   >
   > Kata sandi yang Anda buat harus memenuhi kebijakan kata sandi akun.

7. Untuk mengharuskan pengguna membuat kata sandi baru saat masuk, pilih Memer lukan perubahan kata sandi pada saat masuk berikutnya.
8. Untuk mengharuskan pengguna segera menggunakan kata sandi baru, pilih Cab ut sesi konsol aktif. Ini melampirkan kebijakan sebaris ke pengguna IAM yang menolak akses pengguna ke sumber daya jika kredensialnya lebih tua dari waktu yang ditentukan oleh kebijakan.
9. Pilih Reset kata sandi
10. Dialog kata sandi Konsol memberi tahu Anda bahwa Anda telah mengaktifkan kata sandi baru pengguna. Untuk melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna, pilih Tampilkan di kotak dialog kata sandi konsol. Pilih Unduh file.csv untuk mengunduh file dengan kredenSIAL pengguna.

    > **Important**
    >
    > Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa akses konsol telah diaktifkan.

Untuk mengubah kata sandi bagi pengguna IAM (konsol)

Gunakan prosedur ini untuk memperbarui kata sandi yang terkait dengan nama pengguna.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.

2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Cari IAM.
3. Di panel navigasi, pilih Pengguna.
4. Pilih nama pengguna yang kata sandinya ingin Anda ubah.
5. Pilih tab Kredensibilitas keamanan, lalu di bawah Mas uk ke konsol, pilih Kel ola akses konsol.
6. Di kotak dialog Kel ola akses konsol, pilih Reset kata sandi, lalu pilih apakah IAM akan membuat kata sandi atau membuat kata sandi khusus:
   - Untuk membuat IAM membuat kata sandi, pilih Kata sandi yang dibuat otomatis.
   - Untuk membuat kata sandi kustom, pilih Kata sandi kustom, dan ketik kata sandi.

   > **Note**
   >
   > Kata sandi yang Anda buat harus memenuhi kebijakan kata sandi akun.

7. Untuk mengharuskan pengguna membuat kata sandi baru saat masuk, pilih Memer lukan perubahan kata sandi pada saat masuk berikutnya.
8. Untuk mengharuskan pengguna segera menggunakan kata sandi baru, pilih Cab ut sesi konsol aktif. Ini melampirkan kebijakan sebaris ke pengguna IAM yang menolak akses pengguna ke sumber daya jika kredensialnya lebih tua dari waktu yang ditentukan oleh kebijakan.
9. Pilih Reset kata sandi
10. Dialog kata sandi Konsol memberi tahu Anda bahwa Anda telah mengaktifkan kata sandi baru pengguna. Untuk melihat kata sandi sehingga Anda dapat membagikannya dengan pengguna, pilih Tampilkan di kotak dialog kata sandi konsol. Pilih Unduh file.csv untuk mengunduh file dengan kredenSIAL pengguna.

    > **Important**
    >
    > Untuk alasan keamanan, Anda tidak dapat mengakses kata sandi setelah menyelesaikan langkah ini, tetapi Anda dapat membuat kata sandi baru kapan saja.

Konsol menampilkan pesan status yang memberi tahu Anda bahwa akses konsol telah diperbarui.

Untuk menghapus (menonaktifkan) kata sandi pengguna IAM (konsol)

Gunakan prosedur ini untuk menghapus kata sandi yang terkait dengan nama pengguna, menghapus akses konsol untuk pengguna.

> **Important**
>
> Anda dapat mencegah pengguna IAM mengakses Konsol Manajemen AWS dengan menghapus kata sandi mereka. Ini mencegah mereka masuk ke Konsol Manajemen AWS menggunakan kredenSIAL masuk mereka. Hal ini tidak mengubah izin mereka atau mencegah mereka mengakses konsol menggunakan peran yang diasumsikan. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Alat untuk Windows PowerShell, AWS API, atau Aplikasi Sel AWS uler Konsol.

Console

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna.
2. Pada halaman Beranda Konsol IAM, di panel navigasi kiri, masukkan kueri Anda di kotak teks Cari IAM.
3. Di panel navigasi, pilih Pengguna.
4. Pilih nama pengguna yang kata sandinya ingin Anda hapus.
5. Pilih tab Kredensibilitas keamanan, lalu di bawah Mas uk ke konsol, pilih Kel ola akses konsol.
6. Untuk mengharuskan pengguna berhenti menggunakan konsol segera, pilih Cab ut sesi konsol aktif. Ini melampirkan kebijakan sebaris ke pengguna IAM yang menolak akses pengguna ke sumber daya jika kredensialnya lebih tua dari waktu yang ditentukan oleh kebijakan.
7. Pilih N onaktifkan akses

Konsol menampilkan pesan status yang memberi tahu Anda bahwa akses konsol telah dinonaktifkan.

Membuat, mengubah, atau menghapus kata sandi pengguna IAM (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengelola kata sandi untuk pengguna IAM Anda.

Untuk membuat kata sandi (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. Untuk membuat kata sandi, jalankan perintah ini: aws iam create-login-profile

Untuk mengubah kata sandi pengguna (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. Untuk mengubah kata sandi, jalankan perintah ini: aws iam update-login-profile

Untuk menghapus (menonaktifkan) kata sandi pengguna (AWS CLI)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: aws iam get-login-profile
2. (Opsional) Untuk menentukan kapan kata sandi terakhir digunakan, jalankan perintah ini: aws iam get-user
3. Untuk menghapus kata sandi, jalankan perintah ini: aws iam delete-login-profile

> **Important**
>
> Ketika Anda menghapus kata sandi pengguna, pengguna tidak bisa lagi masuk ke Konsol Manajemen AWS. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Tools for Windows PowerShell, atau panggilan fungsi AWS API. Saat Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus pengguna dari Anda Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat Menghapus pengguna IAM (AWS CLI).

Untuk mencabut sesi konsol aktif pengguna sebelum waktu tertentu (AWS CLI)

1. Untuk menyematkan kebijakan sebaris yang mencabut sesi konsol aktif pengguna IAM sebelum waktu tertentu, gunakan kebijakan sebaris berikut dan jalankan perintah ini: aws iam put-user-policy

   Kebijakan sebaris ini menolak semua izin dan menyertakan kunci `aw: TokenIssueTime` kondisi. Ini mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan dalam `Condition` elemen kebijakan sebaris. Ganti nilai kunci `aws:TokenIssueTime` kondisi dengan nilai Anda sendiri.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Opsional) Untuk mencantumkan nama kebijakan sebaris yang disematkan pada pengguna IAM, jalankan perintah ini: aws iam list-user-policies
3. (Opsional) Untuk melihat kebijakan inline bernama yang disematkan di pengguna IAM, jalankan perintah ini: aws iam get-user-policy

Membuat, mengubah, atau menghapus kata sandi pengguna IAM (AWS API)

Anda dapat menggunakan AWS API untuk mengelola kata sandi untuk pengguna IAM Anda.

Untuk membuat kata sandi (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, hubungi operasi ini: GetLoginProfile

2. Untuk membuat kata sandi, hubungi operasi ini: CreateLoginProfile

Untuk mengubah kata sandi pengguna (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, hubungi operasi ini: GetLoginProfile
2. Untuk mengubah kata sandi, hubungi operasi ini: UpdateLoginProfile

Untuk menghapus (menonaktifkan) kata sandi pengguna (AWS API)

1. (Opsional) Untuk menentukan apakah pengguna memiliki kata sandi, jalankan perintah ini: GetLoginProfile
2. (Opsional) Untuk menentukan kapan kata sandi terakhir digunakan, jalankan perintah ini: GetUser
3. Untuk menghapus kata sandi, jalankan perintah ini: DeleteLoginProfile

> **Important**
>
> Ketika Anda menghapus kata sandi pengguna, pengguna tidak bisa lagi masuk ke Konsol Manajemen AWS. Jika pengguna memiliki kunci akses aktif, mereka terus berfungsi dan mengizinkan akses melalui AWS CLI, Tools for Windows PowerShell, atau panggilan fungsi AWS API. Saat Anda menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API untuk menghapus pengguna dari Anda Akun AWS, Anda harus terlebih dahulu menghapus kata sandi menggunakan operasi ini. Untuk informasi selengkapnya, lihat Menghapus pengguna IAM (AWS CLI).

Untuk mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan (AWS API)

1. Untuk menyematkan kebijakan sebaris yang mencabut sesi konsol aktif pengguna IAM sebelum waktu yang ditentukan, gunakan kebijakan sebaris berikut dan jalankan perintah ini: PutUserPolicy

   Kebijakan sebaris ini menolak semua izin dan menyertakan kunci `aw: TokenIssueTime` kondisi. Ini mencabut sesi konsol aktif pengguna sebelum waktu yang ditentukan dalam

`Condition` elemen kebijakan sebaris. Ganti nilai kunci `aws:TokenIssueTime` kondisi dengan nilai Anda sendiri.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Opsional) Untuk mencantumkan nama kebijakan sebaris yang disematkan pada pengguna IAM, jalankan perintah ini: ListUserPolicies
3. (Opsional) Untuk melihat kebijakan inline bernama yang disematkan di pengguna IAM, jalankan perintah ini: GetUserPolicy

Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri

> **Note**
>
> Pengguna dengan identitas federasi akan menggunakan proses yang ditentukan oleh penyedia identitas mereka untuk mengubah kata sandi mereka. Sebagai praktik terbaik, mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredenSIAL sementara.

Anda dapat memberikan izin kepada pengguna IAM untuk mengubah kata sandi mereka sendiri untuk masuk ke Konsol Manajemen AWS. Anda dapat melakukannya dengan salah satu dari dua cara berikut:

- Izinkan semua pengguna IAM di akun untuk mengubah kata sandi mereka sendiri.

- Izinkan pengguna IAM terpilih untuk mengubah kata sandi mereka sendiri.. Dalam skenario ini, Anda menonaktifkan opsi bagi semua pengguna untuk mengubah kata sandi mereka sendiri menggunakan kebijakan IAM untuk memberikan izin hanya kepada beberapa pengguna. Pendekatan ini memungkinkan pengguna untuk mengubah kartu sandi mereka sendiri dan secara opsional kredensial lainnya seperti kunci akses mereka sendiri.

> **Important**
>
> Kami sarankan Anda menetapkan kebijakan kata sandi khusus yang memerlukan pengguna IAM untuk membuat kata sandi yang kuat.

Untuk mengizinkan semua pengguna IAM mengubah kata sandi mereka sendiri

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pilih Edit.
4. Pilih Kustom untuk menggunakan kebijakan kata sandi khusus.
5. Pilih Izinkan pengguna mengubah kata sandi mereka sendiri, lalu pilih Simpan perubahan. Ini memungkinkan semua pengguna di akun mengakses `iam:ChangePassword` tindakan hanya untuk pengguna mereka dan `iam:GetAccountPasswordPolicy` tindakan.
6. Berikan petunjuk berikut kepada pengguna untuk mengubah kata sandi mereka: Bagaimana pengguna IAM mengubah kata sandi mereka sendiri.

AWS CLI

Jalankan perintah berikut:

- `aws iam update-account-password-policy`

API

Untuk memperbarui kebijakan kata sandi akun, panggil operasi berikut:

- UpdateAccountPasswordPolicy

Untuk mengizinkan pengguna IAM yang dipilih mengubah kata sandi mereka sendiri

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengaturan akun.
3. Di bagian Kebijakan kata sandi, pastikan bahwa Perbolehkan pengguna mengubah kata sandi mereka sendiri tidak dipilih. Jika kotak centang ini dipilih, semua pengguna dapat mengubah kata sandi mereka sendiri. (Lihat prosedur sebelumnya.)
4. Buat pengguna yang seharusnya diperbolehkan untuk mengubah kata sandi mereka sendiri, jika mereka belum ada. Lihat perinciannya di Buat pengguna IAM di Akun AWS.
5. (Opsional) Buat grup IAM untuk pengguna yang seharusnya diizinkan mengubah kata sandi mereka, lalu tambahkan pengguna dari langkah sebelumnya ke grup. Lihat perinciannya di Grup pengguna IAM.
6. Tetapkan kebijakan berikut kepada grup. Untuk informasi selengkapnya, lihat Kelola kebijakan IAM.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Kebijakan ini memberikan akses ke ChangePassword tindakan, yang memungkinkan pengguna hanya mengubah kata sandi mereka sendiri dari konsol, AWS CLI, Alat untuk Windows PowerShell, atau API. Juga memberikan akses ke tindakan GetAccountPasswordPolicy, yang memungkinkan pengguna melihat kebijakan kata sandi saat ini; izin ini diperlukan agar pengguna dapat melihat kebijakan kata sandi akun di halaman Ubah kata sandi. Pengguna harus diizinkan membaca kebijakan kata sandi saat ini untuk memastikan bahwa kata sandi yang diubah memenuhi persyaratan kebijakan.

7. Berikan petunjuk berikut kepada pengguna untuk mengubah kata sandi mereka: Bagaimana pengguna IAM mengubah kata sandi mereka sendiri.

Untuk informasi lebih lanjut

Untuk informasi selengkapnya tentang mengelola kredensial, lihat topik berikut:

- Izinkan pengguna IAM untuk mengubah kata sandi mereka sendiri
- Kata sandi pengguna di AWS
- Menetapkan kebijakan kata sandi akun untuk pengguna IAM
- Kelola kebijakan IAM
- Bagaimana pengguna IAM mengubah kata sandi mereka sendiri

Bagaimana pengguna IAM mengubah kata sandi mereka sendiri

Jika Anda telah diberikan izin untuk mengubah kata sandi pengguna IAM Anda sendiri, Anda dapat menggunakan halaman khusus Konsol Manajemen AWS untuk melakukan ini. Anda juga dapat menggunakan AWS CLI atau AWS API.

Topik

- Izin diperlukan
- Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)
- Bagaimana pengguna IAM mengubah kata sandi mereka sendiri (AWS CLI atau AWS API)

Izin diperlukan

Untuk mengubah kata sandi untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut: AWS: Memungkinkan pengguna IAM untuk mengubah kata sandi konsol mereka sendiri di halaman kredensi Keamanan.

Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

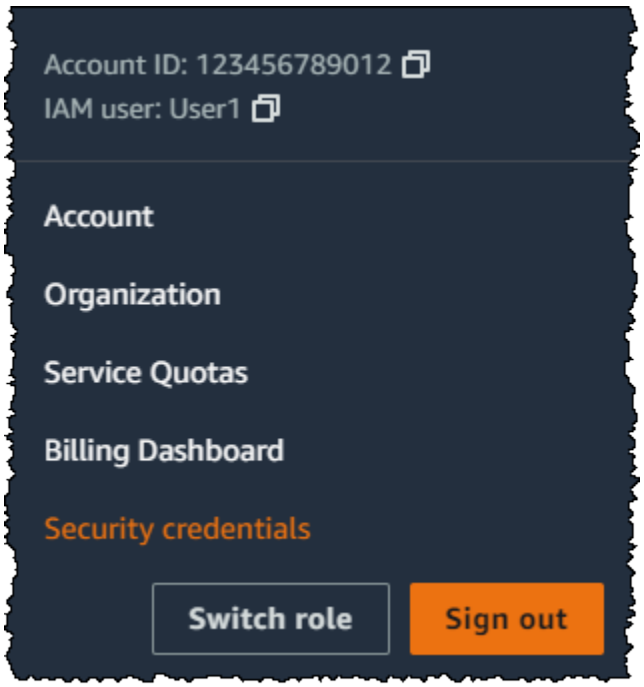
Prosedur berikut menjelaskan bagaimana pengguna IAM dapat menggunakan Konsol Manajemen AWS untuk mengubah kata sandi mereka sendiri.

Untuk mengubah kata sandi pengguna IAM Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun IAM Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada tab Kredensial AWS IAM, pilih Perbarui kata sandi.
4. Untuk Kata sandi saat ini, masukkan kata sandi Anda saat ini. Masukkan kata sandi baru untuk Kata sandi baru dan Konfirmasi kata sandi baru. Kemudian pilih Perbarui kata sandi.

> **Note**
>
> Kata sandi baru harus memenuhi persyaratan kebijakan kata sandi akun. Untuk informasi selengkapnya, lihat Menetapkan kebijakan kata sandi akun untuk pengguna IAM.

Bagaimana pengguna IAM mengubah kata sandi mereka sendiri (AWS CLI atau AWS API)

Prosedur berikut menjelaskan bagaimana pengguna IAM dapat menggunakan AWS CLI atau AWS API untuk mengubah kata sandi mereka sendiri.

Untuk mengubah kata sandi IAM Anda sendiri, gunakan cara berikut:

- AWS CLI: `aws iam change-password`
- AWS API: `ChangePassword`

Kelola access key untuk pengguna IAM

> **Important**
>
> Sebagai praktik terbaik, gunakan kredensi keamanan sementara (seperti peran IAM) alih-alih membuat kredensi jangka panjang seperti kunci akses. Sebelum membuat kunci akses, tinjau alternatif untuk kunci akses jangka panjang.

Kunci akses adalah kredensi jangka panjang untuk pengguna IAM atau. Pengguna root akun AWS Anda dapat menggunakan kunci akses untuk menandatangani permintaan terprogram ke AWS CLI atau AWS API (secara langsung atau menggunakan AWS SDK). Untuk informasi selengkapnya, lihat Akses terprogram dengan AWS kredensial keamanan.

Access key terdiri dari dua bagian: access key ID (misalnya, `AKIAIOSFODNN7EXAMPLE`) dan secret access key (misalnya, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Anda harus menggunakan ID kunci akses dan kunci akses rahasia bersama-sama untuk mengautentikasi permintaan Anda.

Saat Anda membuat pasangan access key, simpan access key ID dan secret access key di lokasi yang aman. Kunci akses rahasia hanya dapat diambil pada saat Anda membuatnya. Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Untuk instruksi lebih lanjut, lihatPerbarui kunci akses.

Anda dapat memiliki maksimal dua kunci akses per pengguna.

> **Important**
>
> Pengguna IAM dengan kunci akses adalah risiko keamanan akun. Kelola kunci akses Anda dengan aman. Jangan berikan kunci akses Anda kepada pihak yang tidak berwenang, bahkan untuk membantu menemukan pengenal akun Anda. Dengan melakukan tindakan ini, Anda mungkin memberi seseorang akses permanen ke akun Anda.
> Saat bekerja dengan kunci akses, perhatikan hal-hal berikut:
>
> - JANGAN gunakan kredensi root akun Anda untuk membuat kunci akses.
> - JANGAN menaruh kunci akses atau informasi kredensi dalam file aplikasi Anda.
> - JANGAN sertakan file yang berisi kunci akses atau informasi kredensi di area proyek Anda.

- Kunci akses atau informasi kredensi yang disimpan dalam file AWS kredensial bersama disimpan dalam teks biasa.

Rekomendasi pemantauan

Setelah membuat kunci akses:

- Gunakan AWS CloudTrail untuk memantau penggunaan kunci akses dan mendeteksi upaya akses yang tidak sah. Untuk informasi selengkapnya, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail.
- Siapkan CloudWatch alarm untuk memberi tahu administrator untuk upaya akses yang ditolak untuk membantu mendeteksi aktivitas berbahaya. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.
- Tinjau, perbarui, dan hapus kunci akses secara teratur sesuai kebutuhan.

Topik berikut merinci tugas manajemen yang terkait dengan kunci akses.

Topik

- Mengontrol penggunaan kunci akses dengan melampirkan kebijakan sebaris ke pengguna IAM
- Izin diperlukan untuk mengelola kunci akses
- Bagaimana pengguna IAM dapat mengelola kunci akses mereka sendiri
- Bagaimana administrator IAM dapat mengelola kunci akses pengguna IAM
- Perbarui kunci akses
- Kunci akses aman

Mengontrol penggunaan kunci akses dengan melampirkan kebijakan sebaris ke pengguna IAM

Sebagai praktik terbaik, kami menyarankan agar beban kerja menggunakan kredenSIAL sementara dengan peran IAM untuk mengakses. AWS Pengguna IAM dengan kunci akses harus diberi akses hak istimewa terkecil dan mengaktifkan otentikasi multi-faktor (MFA). Untuk informasi selengkapnya tentang mengambil peran IAM, lihatMetode untuk mengambil peran.

Namun, jika Anda membuat uji bukti konsep otomatisasi layanan atau kasus penggunaan jangka pendek lainnya, dan Anda memilih untuk menjalankan beban kerja menggunakan pengguna IAM

dengan kunci akses, kami sarankan Anda menggunakan kondisi kebijakan untuk lebih membatasi akses kredentif pengguna IAM mereka.

Dalam situasi ini, Anda dapat membuat kebijakan terikat waktu yang kedaluwarsa kredenSIAL setelah waktu yang ditentukan atau, jika Anda menjalankan beban kerja dari jaringan aman, Anda dapat menggunakan kebijakan pembatasan IP.

Untuk kedua kasus penggunaan ini, Anda dapat menggunakan kebijakan sebaris yang dilampirkan ke pengguna IAM yang memiliki kunci akses.

Untuk mengonfigurasi kebijakan batas waktu untuk pengguna IAM

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna dan kemudian pilih pengguna untuk kasus penggunaan jangka pendek. Jika Anda belum membuat pengguna, Anda dapat membuat pengguna sekarang.
3. Pada halaman Detail pengguna, pilih tab Izin.
4. Pilih Tambah izin lalu pilih Buat kebijakan sebaris.
5. Di bagian Editor kebijakan, pilih JSON untuk menampilkan editor JSON.
6. Di editor JSON, masukkan kebijakan berikut, ganti nilai untuk stempel waktu dengan tanggal `aws:CurrentTime` dan waktu kedaluwarsa yang Anda inginkan:

   JSON

```
{
"Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
      "DateGreaterThan": {
      "aws:CurrentTime": "2025-03-01T00:12:00Z"
        }
      }
    }
  ]
```

```
}
```

Kebijakan ini menggunakan `Deny` efek untuk membatasi semua tindakan pada semua sumber daya setelah tanggal yang ditentukan. `DateGreaterThan`Kondisi membandingkan waktu saat ini dengan stempel waktu yang Anda tetapkan.

7. Pilih Ber ikutnya untuk melanjutkan ke halaman Review dan buat. Di Detail kebijakan, di bawah Nama kebijakan masukkan nama untuk kebijakan, lalu pilih Buat kebijakan.

Setelah kebijakan dibuat, kebijakan tersebut akan ditampilkan pada tab Izin untuk pengguna. Ketika waktu saat ini lebih besar dari atau sama dengan waktu yang ditentukan dalam kebijakan, pengguna tidak akan lagi memiliki akses ke AWS sumber daya. Pastikan untuk memberi tahu pengembang beban kerja tentang tanggal kedaluwarsa yang Anda tentukan untuk kunci akses ini.

Untuk mengonfigurasi kebijakan pembatasan IP untuk pengguna IAM

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna dan kemudian pilih pengguna yang akan menjalankan beban kerja dari jaringan aman. Jika Anda belum membuat pengguna, Anda dapat membuat pengguna sekarang.
3. Pada halaman Detail pengguna, pilih tab Izin.
4. Pilih Tambah izin lalu pilih Buat kebijakan sebaris.
5. Di bagian Editor kebijakan, pilih JSON untuk menampilkan editor JSON.
6. Salin kebijakan IAM berikut ke editor JSON, dan ubah alamat IPv4 atau IPv6 publik, atau rentang sesuai kebutuhan Anda. Anda dapat menggunakan https://checkip.amazonaws.com/ untuk menentukan alamat IP publik Anda saat ini. Anda dapat menentukan alamat IP individual, atau rentang alamat IP menggunakan notasi garis miring. Untuk informasi selengkapnya, lihat aws:SourceIp.

> **Note**
>
> Alamat IP tidak boleh dikaburkan oleh VPN atau server proxy.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"IpRestrictionIAMPolicyForIAMUser",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Contoh kebijakan ini menolak penggunaan kunci akses pengguna IAM dengan kebijakan ini diterapkan, kecuali permintaan berasal dari jaringan (ditentukan dalam notasi CIDR) “203.0.113.0/24”, “2001:DB8:1234:5678: :/64”, atau alamat IP spesifik “203.0.114.1”

7. Pilih Ber ikutnya untuk melanjutkan ke halaman Review dan buat. Di Detail kebijakan, di bawah Nama kebijakan masukkan nama untuk kebijakan, lalu pilih Buat kebijakan.

Setelah kebijakan dibuat, kebijakan tersebut akan ditampilkan pada tab Izin untuk pengguna.

Anda juga dapat menerapkan kebijakan ini sebagai kebijakan kontrol layanan (SCP) di beberapa AWS akun di AWS Organizations, sebaiknya gunakan kondisi tambahan, agar pernyataan kebijakan

ini hanya berlaku `aws:PrincipalArn` untuk pengguna IAM dalam AWS akun yang tunduk pada SCP ini. Kebijakan berikut mencakup pembaruan tersebut:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IpRestrictionServiceControlPolicyForIAMUsers",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:user/*"
        }
      }
    }
  ]
}
```

Izin diperlukan untuk mengelola kunci akses

> **Note**
>
> `iam:TagUser`adalah izin opsional untuk menambahkan dan mengedit deskripsi untuk kunci akses. Untuk informasi selengkapnya, lihat Menandai pengguna IAM

Untuk membuat kunci akses untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Untuk memperbarui kunci akses untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
```

```
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Bagaimana pengguna IAM dapat mengelola kunci akses mereka sendiri

Administrator IAM dapat memberikan izin kepada pengguna IAM untuk mengelola sendiri kunci akses mereka dengan melampirkan kebijakan yang dijelaskan di dalamnya. Izin diperlukan untuk mengelola kunci akses

Dengan izin ini, pengguna IAM dapat menggunakan prosedur berikut untuk membuat, mengaktifkan, menonaktifkan, dan menghapus kunci akses yang terkait dengan nama pengguna mereka.

Topik

- Buat kunci akses untuk Anda sendiri (konsol)
- Nonaktifkan kunci akses Anda (konsol)
- Aktifkan kunci akses Anda (konsol)
- Hapus kunci akses Anda (konsol)

Buat kunci akses untuk Anda sendiri (konsol)

Jika Anda telah diberikan izin yang sesuai, Anda dapat menggunakan tombol Konsol Manajemen AWS untuk membuat kunci akses untuk diri Anda sendiri.

Untuk membuat kunci akses Anda sendiri (konsol)

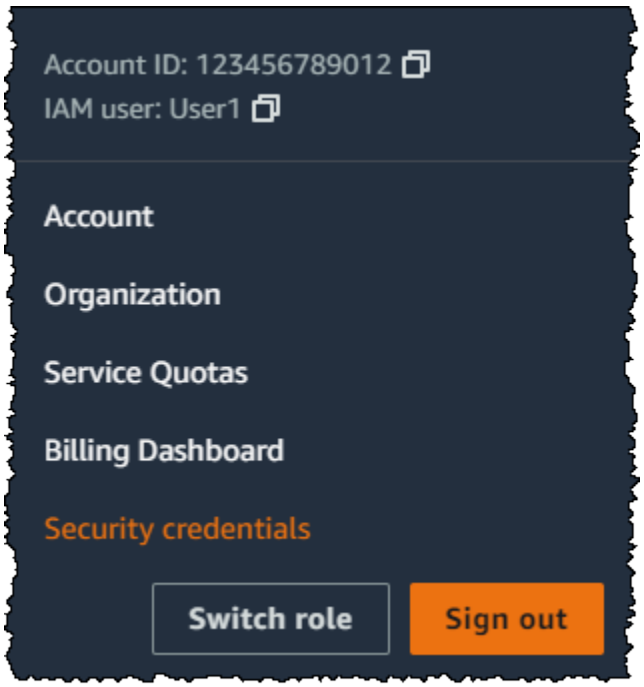
1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

> **Note**
>
> Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS

akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Pada bagian Access key, pilih Buat access key. Jika Anda sudah memiliki dua kunci akses, tombol ini dinonaktifkan dan Anda harus menghapus kunci akses sebelum Anda dapat membuat yang baru.
4. Pada halaman praktik terbaik & alternatif kunci Access, pilih kasus penggunaan Anda untuk mempelajari opsi tambahan yang dapat membantu Anda menghindari pembuatan kunci akses jangka panjang. Jika Anda menentukan bahwa kasus penggunaan Anda masih memerlukan kunci akses, pilih Lainnya dan kemudian pilih Berikutnya.
5. (Opsional) Tetapkan nilai tag deskripsi untuk kunci akses. Ini menambahkan pasangan nilai kunci tag ke pengguna IAM Anda. Ini dapat membantu Anda mengidentifikasi dan memperbarui kunci akses nanti. Kunci tag diatur ke id kunci akses. Nilai tag diatur ke deskripsi kunci akses yang Anda tentukan. Setelah selesai, pilih Buat kunci akses.
6. Pada halaman Ambil kunci akses, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda, atau Unduh file.csv. Ini adalah satu-satunya kesempatan untuk menyimpan secret access key Anda. Setelah menyimpan kunci akses rahasia di lokasi yang aman, pilih Selesai.

Nonaktifkan kunci akses Anda (konsol)

Jika Anda telah diberikan izin yang sesuai, Anda dapat menggunakan tombol Konsol Manajemen AWS untuk menonaktifkan kunci akses Anda.

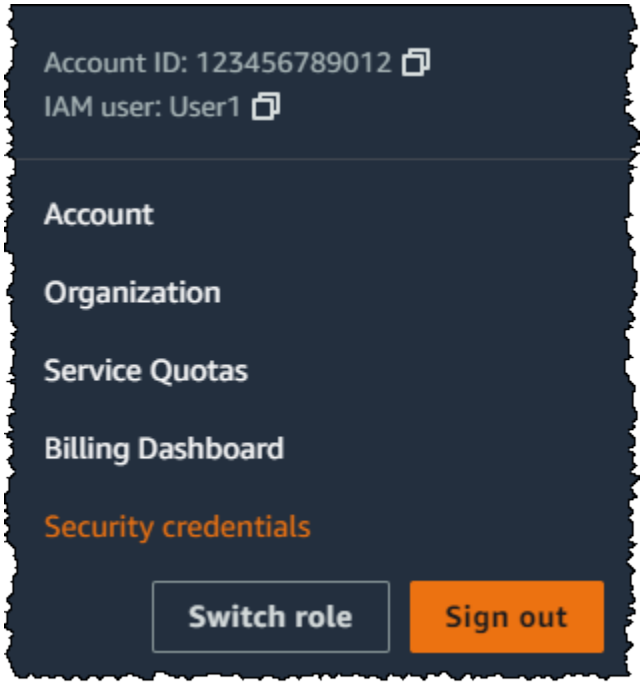
Untuk menonaktifkan kunci akses

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Di bagian Kunci akses temukan kunci yang ingin Anda nonaktifkan, lalu pilih Tindakan, lalu pilih Nonaktifkan. Saat diminta konfirmasi, pilih Deactivate (Nonaktifkan). Access key yang dinonaktifkan masih diperhitungkan dalam batas dua access key Anda.

Aktifkan kunci akses Anda (konsol)

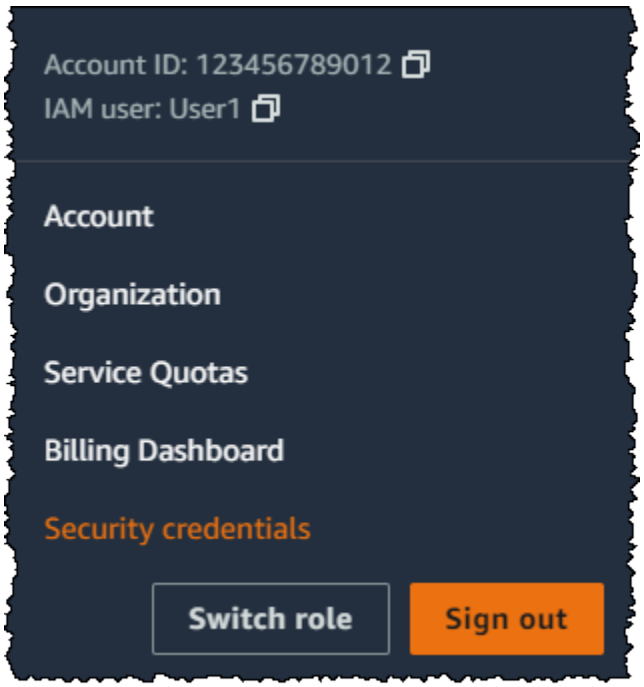
Jika Anda telah diberikan izin yang sesuai, Anda dapat menggunakan tombol Konsol Manajemen AWS untuk mengaktifkan kunci akses Anda.

Untuk mengaktifkan kunci akses

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Di bagian Kunci akses, temukan kunci untuk mengaktifkan, lalu pilih Tindakan, lalu pilih Aktifkan.

Hapus kunci akses Anda (konsol)

Jika Anda telah diberikan izin yang sesuai, Anda dapat menggunakan tombol Konsol Manajemen AWS untuk menghapus kunci akses Anda.

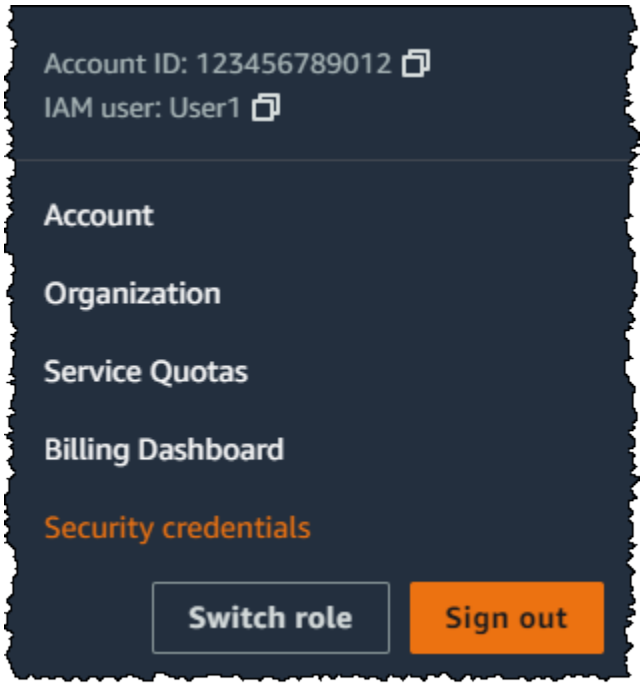
Untuk menghapus kunci akses saat Anda tidak lagi membutuhkannya

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna IAM dan informasi akun Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun Anda untuk diarahkan ke halaman login pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.



3. Di bagian Kunci akses, cari kunci yang ingin Anda hapus, lalu pilih Tindakan, lalu pilih Hapus. Ikuti instruksi dalam dialog untuk pertama Nonaktifkan dan kemudian konfirmasikan penghapusan. Sebaiknya verifikasi bahwa access key tidak lagi digunakan sebelum Anda menghapusnya secara permanen.

Bagaimana administrator IAM dapat mengelola kunci akses pengguna IAM

Administrator IAM dapat membuat, mengaktifkan, menonaktifkan, dan menghapus kunci akses yang terkait dengan pengguna IAM individu. Mereka juga dapat mencantumkan pengguna IAM di akun yang memiliki kunci akses dan menemukan pengguna IAM mana yang memiliki kunci akses tertentu.

Topik

- Untuk membuat kunci akses untuk pengguna IAM
- Untuk menonaktifkan kunci akses untuk pengguna IAM
- Untuk mengaktifkan kunci akses untuk pengguna IAM
- Untuk menghapus kunci akses untuk pengguna IAM
- Untuk membuat daftar kunci akses untuk pengguna IAM
- Untuk menampilkan semua ID kunci akses bagi pengguna di akun Anda
- Untuk menggunakan ID kunci akses untuk menemukan pengguna

- Untuk menemukan penggunaan terbaru dari ID kunci akses

Untuk membuat kunci akses untuk pengguna IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Pada tab Security credentials, di bagian Access keys, pilih Create Access Key.

   Jika tombol dinonaktifkan, maka Anda harus menghapus salah satu kunci yang ada sebelum Anda dapat membuat yang baru.
5. Pada halaman Praktik & Alternatif Terbaik Kunci Akses, tinjau praktik dan alternatif terbaik. Pilih kasus penggunaan Anda untuk mempelajari opsi tambahan yang dapat membantu Anda menghindari pembuatan kunci akses jangka panjang.
6. Jika Anda menentukan bahwa kasus penggunaan Anda masih memerlukan kunci akses, pilih Lainnya dan kemudian pilih Berikutnya.
7. (Opsional) Pada halaman Setel tag deskripsi, Anda dapat menambahkan tag deskripsi ke kunci akses untuk membantu melacak kunci akses Anda. Pilih Buat tombol akses.
8. Pada halaman Retrieve access key, pilih Tampilkan untuk mengungkapkan nilai kunci akses rahasia pengguna Anda.
9. Untuk menyimpan ID kunci akses dan kunci akses rahasia ke `.csv` file ke lokasi aman di komputer Anda, pilih tombol Unduh file.csv.

   > **Important**
   >
   > Ini adalah satu-satunya waktu Anda untuk melihat atau mengunduh kunci akses yang baru dibuat dan Anda tidak dapat memulihkannya. Pastikan Anda menjaga kunci akses Anda dengan aman.

Saat Anda membuat kunci akses untuk pengguna Anda, key pair tersebut aktif secara default, dan pengguna Anda dapat langsung menggunakan pasangan tersebut.

AWS CLI

Jalankan perintah berikut:

- `aws iam create-access-key`

API

Panggil operasi berikut:

- `CreateAccessKey`

Untuk menonaktifkan kunci akses untuk pengguna IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Pada tab Security credentials, di bagian Access keys, pilih menu drop-down Actions, lalu pilih Nonaktifkan.
5. Di kotak dialog Nonaktifkan, konfirmasikan bahwa Anda ingin menonaktifkan kunci akses dengan memilih Nonaktifkan

Setelah kunci akses dinonaktifkan, itu tidak dapat lagi digunakan oleh panggilan API. Anda dapat mengaktifkannya lagi jika diperlukan.

AWS CLI

Jalankan perintah berikut:

- `aws iam update-access-key`

API

Panggil operasi berikut:

- `UpdateAccessKey`

Untuk mengaktifkan kunci akses untuk pengguna IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Pada tab Security credentials, di bagian Access keys, pilih menu drop-down Actions, lalu pilih Activate.

Setelah kunci akses diaktifkan, itu dapat digunakan oleh panggilan API. Anda dapat menonaktifkannya lagi jika diperlukan.

AWS CLI

Jalankan perintah berikut:

- `aws iam update-access-key`

API

Panggil operasi berikut:

- `UpdateAccessKey`

Untuk menghapus kunci akses untuk pengguna IAM

Setelah kunci akses dinonaktifkan, jika tidak lagi diperlukan, hapus.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Pada tab Security credentials, di bagian Access keys, pilih menu drop-down Actions untuk tombol akses tidak aktif, lalu pilih Delete.

5. Di kotak dialog Hapus, konfirmasikan bahwa Anda ingin menghapus kunci akses dengan memasukkan ID kunci akses di bidang input teks dan kemudian memilih Hapus.

Setelah kunci akses dihapus, itu tidak dapat dipulihkan.

AWS CLI

Jalankan perintah berikut:

- `aws iam delete-access-key`

API

Panggil operasi berikut:

- `DeleteAccessKey`

Untuk membuat daftar kunci akses untuk pengguna IAM

Anda dapat melihat daftar ID kunci akses yang terkait dengan pengguna IAM.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Pada tab Security credentials, bagian Access keys mencantumkan ID kunci akses untuk pengguna termasuk status setiap kunci yang ditampilkan.

> **Note**
>
> Hanya access key ID pengguna yang terlihat. Secret access key hanya dapat diambil kembali saat kunci dibuat.

Setiap pengguna IAM dapat memiliki dua kunci akses.

AWS CLI

Jalankan perintah berikut:

- `aws iam list-access-keys`

API

Panggil operasi berikut:

- `ListAccessKeys`

Untuk menampilkan semua ID kunci akses bagi pengguna di akun Anda

Anda dapat melihat daftar ID kunci akses untuk pengguna di Anda Akun AWS.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna untuk membuka halaman detail pengguna.
4. Jika perlu, tambahkan kolom access key ID pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:
   a. Di atas tabel di paling kanan, pilih ikon Preferensi ( ).
   b. Di kotak dialog Preferensi, di bawah Pilih kolom yang terlihat, aktifkan ID kunci akses.
   c. Pilih Konfirmasi untuk kembali ke daftar pengguna. Daftar diperbarui untuk menyertakan ID kunci akses.
5. Kolom ID kunci Access menunjukkan status setiap kunci akses, diikuti dengan ID-nya; misalnya, **`Active - AKIAIOSFODNN7EXAMPLE`**atau **`Inactive - AKIAI44QH8DHBEXAMPLE`**.

   Anda dapat menggunakan informasi ini untuk melihat dan menyalin ID kunci akses untuk pengguna dengan satu atau dua kunci akses. Kolom ditampilkan **`-`**untuk pengguna tanpa kunci akses.

> **Note**
>
> Secret access key hanya dapat diambil kembali saat kunci dibuat.

Setiap pengguna IAM dapat memiliki dua kunci akses.

Untuk menggunakan ID kunci akses untuk menemukan pengguna

Anda dapat menggunakan ID kunci akses untuk menemukan pengguna di akun Anda Akun AWS.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, di kotak pencarian, masukkan ID kunci akses, misalnyaAKIAI44QH8DHBEXAMPLE.
3. Pengguna IAM yang dikaitkan dengan ID kunci akses muncul di panel navigasi. Pilih nama pengguna untuk membuka halaman detail pengguna.

Untuk menemukan penggunaan terbaru dari ID kunci akses

Penggunaan kunci akses terbaru ditampilkan dalam daftar pengguna di halaman pengguna IAM, di halaman detail pengguna, dan merupakan bagian dari laporan kredensi.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Dalam daftar pengguna, lihat kolom Access key last used.

   Jika kolom tidak ditampilkan, pilih ikon Preferensi
   ( )
   dan di bawah Pilih kolom yang terlihat, aktifkan Kunci akses terakhir digunakan untuk menampilkan kolom.

3. (opsional) Di panel navigasi, di bawah Laporan akses, pilih Laporan kredenal untuk mengunduh laporan yang menyertakan kunci akses informasi yang terakhir digunakan untuk semua pengguna IAM di akun Anda.
4. (opsional) Pilih pengguna IAM untuk melihat detail pengguna. Bagian Ringkasan mencakup ID kunci akses, statusnya, dan kapan terakhir kali digunakan.

AWS CLI

Jalankan perintah berikut:

- `aws iam get-access-key-last-used`

API

Panggil operasi berikut:

- `GetAccessKeyLastUsed`

Perbarui kunci akses

Sebagai praktik terbaik keamanan, kami menyarankan Anda memperbarui kunci akses pengguna IAM saat diperlukan, seperti ketika karyawan meninggalkan perusahaan Anda. Pengguna IAM dapat memperbarui kunci akses mereka sendiri jika mereka telah diberikan izin yang diperlukan.

Untuk detail tentang memberikan izin kepada pengguna IAM untuk memperbarui kunci akses mereka sendiri, lihatAWS: Memungkinkan pengguna IAM untuk mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan. Anda juga dapat menerapkan kebijakan kata sandi ke akun Anda untuk mengharuskan semua pengguna IAM Anda memperbarui kata sandi mereka secara berkala dan seberapa sering mereka harus melakukannya. Untuk informasi selengkapnya, lihat Menetapkan kebijakan kata sandi akun untuk pengguna IAM.

> **Note**
>
> Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Kunci akses rahasia hanya dapat diambil pada saat Anda membuatnya. Gunakan prosedur ini untuk menonaktifkan dan kemudian mengganti kunci akses yang hilang dengan kredenSIAL baru.

Topik

- Memperbarui kunci akses pengguna IAM (konsol)
- Memperbarui kunci akses (AWS CLI)
- Memperbarui kunci akses (AWS API)

Memperbarui kunci akses pengguna IAM (konsol)

Anda dapat memperbarui kunci akses dari Konsol Manajemen AWS.

Untuk memperbarui kunci akses untuk pengguna IAM tanpa mengganggu aplikasi Anda (konsol)

1. Meskipun access key pertama masih aktif, buat access key kedua.

   a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   b. Di panel navigasi, pilih Pengguna.

   c. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan.

   d. Pada bagian Access key, pilih Buat access key. Pada halaman Praktik terbaik & alternatif kunci Akses, pilih Lainnya, lalu pilih Ber ikutnya.

   e. (Opsional) Tetapkan nilai tag deskripsi untuk kunci akses untuk menambahkan pasangan kunci-nilai tag ke pengguna IAM ini. Ini dapat membantu Anda mengidentifikasi dan memperbarui kunci akses nanti. Kunci tag diatur ke id kunci akses. Nilai tag disetel ke deskripsi kunci akses yang Anda tentukan. Setelah selesai, pilih Buat kunci akses.

   f. Pada halaman Ambil kunci akses, pilih Tampilkan untuk menampilkan nilai kunci akses rahasia pengguna Anda, atau Unduh file.csv. Ini adalah satu-satunya kesempatan untuk menyimpan secret access key Anda. Setelah Anda menyimpan kunci akses rahasia Anda di lokasi yang aman, pilih Sel esai.

      Saat Anda membuat kunci akses untuk pengguna Anda, pasangan kunci tersebut aktif secara default, dan pengguna Anda dapat menggunakan pasangan tersebut segera. Pada titik ini, pengguna memiliki dua access key aktif.

2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah kunci akses pertama masih digunakan dengan meninjau Informasi terakhir yang digunakan untuk kunci akses tertua. Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.

4. Bahkan jika informasi terakhir digunakan menunjukkan bahwa kunci lama belum pernah digunakan, kami sarankan Anda tidak segera menghapus kunci akses pertama. Sebagai gantinya, pilih T indakan dan kemudian pilih Non aktifkan untuk menonaktifkan kunci akses pertama.
5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Setiap aplikasi dan alat yang masih menggunakan kunci akses asli akan berhenti bekerja pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengaktifkan kembali kunci akses pertama. Lalu kembali ke Step 3 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus kunci akses pertama:

    a. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

    b. Di panel navigasi, pilih Pengguna.

    c. Pilih nama pengguna yang dimaksud, lalu pilih tab Kredensial keamanan.

    d. Di bagian Tombol akses untuk kunci akses yang ingin Anda hapus, pilih Tind akan, lalu pilih H apus. Ikuti instruksi dalam dialog untuk pertama menonaktifkan dan kemudian konfirmasikan penghapusan.

Untuk menentukan kunci akses mana yang perlu diperbarui atau dihapus (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom Usia access key pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:

    a. Di atas tabel di ujung kanan, pilih ikon pengaturan
    ( ).

    b. Di Kelola kolom, pilih Usia access key.

    c. Pilih Tutup untuk kembali ke daftar pengguna.
4. Kolom Usia access key menunjukkan jumlah hari sejak access key aktif tertua dibuat. Anda dapat menggunakan informasi ini untuk menemukan pengguna dengan kunci akses yang

mungkin perlu diperbarui atau dihapus. Kolom menampilkan Tidak ada untuk pengguna yang tidak memiliki access key.

Memperbarui kunci akses (AWS CLI)

Anda dapat memperbarui kunci akses dari AWS Command Line Interface.

Untuk memperbarui kunci akses tanpa mengganggu aplikasi Anda (AWS CLI)

1. Meskipun access key pertama masih aktif, buat access key kedua, yang secara default aktif. Jalankan perintah berikut:

   - `aws iam create-access-key`

   Pada titik ini, pengguna memiliki dua access key aktif.
2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah access key pertama masih digunakan dengan perintah ini:

   - `aws iam get-access-key-last-used`

   Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.
4. Bahkan jika langkah Step 3 tidak menunjukkan penggunaan kunci lama, kami sarankan agar Anda tidak segera menghapus access key pertama. Alih-alih, ubah status access key pertama ke `Inactive` menggunakan perintah ini:

   - `aws iam update-access-key`
5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Setiap aplikasi dan alat yang masih menggunakan kunci akses asli akan berhenti bekerja pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengganti statusnya kembali `Active` untuk mengaktifkan kembali kunci akses pertama. Lalu kembali ke langkah Step 2 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus kunci akses pertama dengan perintah ini:

   - `aws iam delete-access-key`

Memperbarui kunci akses (AWS API)

Anda dapat memperbarui kunci akses menggunakan AWS API.

Untuk memperbarui kunci akses tanpa mengganggu aplikasi Anda (AWS API)

1. Meskipun access key pertama masih aktif, buat access key kedua, yang secara default aktif. Hubungi operasi berikut ini:

   - `CreateAccessKey`

   Pada titik ini, pengguna memiliki dua access key aktif.
2. Perbarui semua aplikasi dan alat untuk menggunakan access key baru.
3. Tentukan apakah access key pertama masih digunakan dengan menghubungi operasi ini:

   - `GetAccessKeyLastUsed`

   Salah satu pendekatannya adalah dengan menunggu beberapa hari kemudian memeriksa access key lama untuk penggunaan apa pun sebelum melanjutkan.
4. Bahkan jika langkah Step 3 tidak menunjukkan penggunaan kunci lama, kami sarankan agar Anda tidak segera menghapus access key pertama. Sebagai gantinya, ubah status kunci akses pertama `Inactive` dengan memanggil operasi ini:

   - `UpdateAccessKey`
5. Gunakan hanya access key baru untuk mengonfirmasi bahwa aplikasi Anda berfungsi. Setiap aplikasi dan alat yang masih menggunakan kunci akses asli akan berhenti bekerja pada saat ini karena mereka tidak lagi memiliki akses ke AWS sumber daya. Jika Anda menemukan aplikasi atau alat seperti itu, Anda dapat mengganti statusnya kembali `Active` untuk mengaktifkan kembali kunci akses pertama. Lalu kembali ke langkah Step 2 dan perbarui aplikasi ini untuk menggunakan kunci baru.
6. Setelah Anda menunggu beberapa waktu untuk memastikan bahwa semua aplikasi dan alat telah diperbarui, Anda dapat menghapus kunci akses pertama dengan memanggil operasi ini:

   - `DeleteAccessKey`

Kunci akses aman

Siapa pun yang memiliki kunci akses Anda memiliki tingkat akses yang sama ke AWS sumber daya Anda dengan yang Anda miliki. Akibatnya, AWS berusaha keras untuk melindungi kunci akses Anda, dan, sesuai dengan model tanggung jawab bersama kami, Anda juga harus melakukannya.

Perluas bagian berikut untuk panduan untuk membantu Anda melindungi kunci akses Anda.

> **Note**
>
> Organisasi Anda mungkin memiliki persyaratan dan kebijakan keamanan yang berbeda dari yang dijelaskan dalam topik ini. Saran yang diberikan di sini dimaksudkan sebagai pedoman umum.

Hapus (atau jangan buat) Pengguna root akun AWS kunci akses

Salah satu cara terbaik untuk melindungi akun Anda adalah dengan tidak memiliki kunci akses untuk Anda Pengguna root akun AWS. Kecuali Anda harus memiliki kunci akses pengguna root (yang jarang terjadi), yang terbaik adalah tidak membuatnya. Sebagai gantinya, buat pengguna administratif AWS IAM Identity Center untuk tugas administrasi harian. Untuk informasi tentang cara membuat pengguna administratif di Pusat Identitas IAM, lihat Mem ulai di Panduan Pengguna Pusat Identitas IAM.

Jika Anda sudah memiliki kunci akses pengguna root untuk akun Anda, kami sarankan hal berikut: Temukan tempat di aplikasi tempat Anda saat ini menggunakan kunci akses (jika ada), dan ganti kunci akses pengguna root dengan kunci akses pengguna IAM. Kemudian nonaktifkan dan hapus kunci akses pengguna root. Untuk informasi selengkapnya tentang cara memperbarui kunci akses, lihat Perbarui kunci akses

Gunakan kredentif keamanan sementara (peran IAM) alih-alih kunci akses jangka panjang

Dalam banyak skenario, Anda tidak memerlukan access key jangka panjang yang tidak pernah kedaluwarsa (seperti yang Anda lakukan dengan pengguna IAM). Sebagai gantinya, Anda dapat membuat IAM role dan membuat kredensial keamanan sementara. Kredensial keamanan sementara terdiri dari access key ID dan secret access key, tetapi mereka juga menyertakan token keamanan yang menunjukkan kapan kredensial kedaluwarsa.

Long-term kunci akses, seperti yang terkait dengan pengguna IAM dan pengguna root, tetap valid hingga Anda mencabutnya secara manual. Namun, kredenSIAL keamanan sementara yang

diperoleh melalui peran IAM dan fitur lainnya akan AWS Security Token Service kedaluwarsa setelah periode waktu yang singkat. Gunakan kredensial keamanan sementara untuk membantu mengurangi risiko Anda jika kredensial terekspos secara tidak sengaja.

Gunakan IAM role dan kredensial keamanan sementara dalam skenario berikut:

- Anda memiliki aplikasi atau AWS CLI skrip yang berjalan di instans Amazon EC2. Jangan gunakan kunci akses langsung di aplikasi Anda. Jangan memberikan access key ke aplikasi, menyematkannya di aplikasi, atau membiarkan aplikasi membaca access key dari sumber mana pun. Sebagai gantinya, tentukan peran IAM yang memiliki izin yang sesuai untuk aplikasi Anda dan luncurkan instans Amazon Elastic Compute Cloud (Amazon EC2) dengan peran untuk EC2. Melakukan hal ini akan mengaitkan IAM role dengan instans Amazon EC2. Praktik ini juga memungkinkan aplikasi untuk mendapatkan kredenSIAL keamanan sementara yang pada gilirannya dapat digunakan untuk melakukan panggilan terprogram. AWS AWS SDK dan AWS Command Line Interface (AWS CLI) dapat memperoleh kredenSIAL sementara dari peran secara otomatis.
- Anda perlu memberikan akses lintas akun. Gunakan IAM role untuk membangun kepercayaan antar akun, lalu berikan izin terbatas kepada pengguna dalam satu akun untuk mengakses akun tepercaya. Untuk informasi selengkapnya, lihat Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM.
- Anda memiliki aplikasi seluler. Jangan menyematkan kunci akses dengan aplikasi, bahkan di penyimpanan terenkripsi. Sebagai gantinya, gunakan Amazon Cognito untuk mengelola identitas pengguna di aplikasi Anda. Layanan ini memungkinkan Anda mengautentikasi pengguna menggunakan Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID Connect (OIDC). Anda kemudian dapat menggunakan penyedia kredensial Amazon Cognito untuk mengelola kredensial yang digunakan aplikasi Anda untuk membuat permintaan ke AWS.
- Anda ingin bergabung AWS dan organisasi Anda mendukung SAML 2.0. Jika Anda bekerja untuk organisasi yang memiliki penyedia identitas yang mendukung SAML 2.0, konfigurasi penyedia untuk menggunakan SAML. Anda dapat menggunakan SAML untuk bertukar informasi otentikasi dengan AWS dan mendapatkan kembali satu set kredenSIAL keamanan sementara. Untuk informasi selengkapnya, lihat Federasi SAML 2.0.
- Anda ingin menyatukan ke dalam AWS dan organisasi Anda memiliki penyimpanan identitas lokal. Jika pengguna dapat melakukan otentikasi di dalam organisasi Anda, Anda dapat menulis aplikasi yang dapat menerbitkan kredenSIAL keamanan sementara untuk akses ke AWS sumber daya. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

- Gunakan kondisi dalam kebijakan IAM untuk hanya mengizinkan akses dari jaringan yang diharapkan. Anda dapat membatasi di mana dan bagaimana kunci akses Anda digunakan dengan menerapkan kebijakan IAM dengan kondisi yang menentukan dan hanya mengizinkan jaringan yang diharapkan, seperti alamat IP publik atau Virtual Private Cloud (VPC). Dengan cara ini Anda tahu kunci akses hanya dapat digunakan dari jaringan yang diharapkan dan dapat diterima.

> **Note**
>
> Apakah Anda menggunakan instans Amazon EC2 dengan aplikasi yang memerlukan akses terprogram ke sumber daya? AWS Jika demikian, gunakan peran IAM untuk EC 2.

Kelola kunci akses pengguna IAM dengan benar

Jika Anda harus membuat kunci akses untuk akses terprogram AWS, buat kunci tersebut untuk pengguna IAM, berikan pengguna hanya izin yang mereka butuhkan.

Perhatikan tindakan pencegahan berikut untuk membantu melindungi kunci akses pengguna IAM:

- Jangan menyematkan kunci akses langsung ke dalam kode. AWS SDK dan Alat Baris AWS Perintah memungkinkan Anda untuk menempatkan kunci akses di lokasi yang diketahui sehingga Anda tidak perlu menyimpannya dalam kode.

  Letakkan access key di salah satu lokasi berikut:

  - File AWS kredenSIAL. AWS SDK dan AWS CLI secara otomatis menggunakan kredenSIAL yang Anda simpan di file AWS kredenSIAL.

    Untuk informasi tentang menggunakan file AWS kredensional, lihat dokumentasi untuk SDK Anda. Contohnya termasuk Tet apkan Kre AWS denSIAL dan Wilayah di Panduan AWS SDK untuk Java Pengembang dan file Konfigurasi dan kreden SIAL di Panduan AWS Command Line Interface Pengguna.

    Untuk menyimpan kredenSIAL untuk AWS SDK untuk .NET dan AWS Tools for Windows PowerShell, kami sarankan Anda menggunakan SDK Store. Untuk informasi selengkapnya, lihat Menggunakan Penyimpanan SDK di Panduan Developer AWS SDK untuk .NET .
  - Variabel lingkungan. Pada sistem multi-tenant, pilih variabel lingkungan pengguna, bukan variabel lingkungan sistem.

Untuk informasi selengkapnya tentang menggunakan variabel lingkungan untuk menyimpan kredensial, lihat Variabel Lingkungan di Panduan Pengguna AWS Command Line Interface .

- Gunakan tombol akses yang berbeda untuk aplikasi yang berbeda. Lakukan ini sehingga Anda dapat mengisolasi izin dan mencabut kunci akses untuk aplikasi individual jika mereka terbuka. Memiliki kunci akses terpisah untuk aplikasi yang berbeda juga menghasilkan entri yang berbeda dalam file AWS CloudTrail log. Konfigurasi ini memudahkan Anda untuk menentukan aplikasi mana yang melakukan tindakan tertentu.
- Perbarui kunci akses saat diperlukan. Jika ada risiko bahwa kunci akses dapat dikompromikan, perbarui kunci akses dan hapus kunci akses sebelumnya. Untuk detailnya, lihat Perbarui kunci akses
- Hapus kunci akses yang tidak digunakan. Jika pengguna keluar dari organisasi Anda, hapus pengguna IAM yang sesuai sehingga pengguna tidak dapat lagi mengakses sumber daya Anda. Untuk mengetahui kapan kunci akses terakhir digunakan, gunakan `GetAccessKeyLastUsed` API (AWS CLI command: `aws iam get-access-key-last-used`).
- Gunakan kredentif sementara dan konfigurasikan otentikasi multi-faktor untuk operasi API Anda yang paling sensitif. Dengan kebijakan IAM, Anda dapat menentukan operasi API mana yang diizinkan untuk dipanggil oleh pengguna. Dalam beberapa kasus, Anda mungkin menginginkan keamanan tambahan yang mengharuskan pengguna diautentikasi dengan AWS MFA sebelum Anda mengizinkan mereka melakukan tindakan yang sangat sensitif. Misalnya, Anda mungkin memiliki kebijakan yang memungkinkan pengguna untuk melakukan Amazon EC2 `RunInstances`, `DescribeInstances`, dan `StopInstances`. Tetapi Anda mungkin ingin membatasi tindakan destruktif seperti `TerminateInstances` dan memastikan bahwa pengguna dapat melakukan tindakan itu hanya jika mereka melakukan otentikasi dengan perangkat AWS MFA. Untuk informasi selengkapnya, lihat Akses API aman dengan MFA.

Akses aplikasi seluler menggunakan AWS kunci akses

Anda dapat mengakses serangkaian AWS layanan dan fitur terbatas menggunakan aplikasi AWS seluler. Aplikasi seluler membantu Anda mendukung respons insiden saat bepergian. Untuk informasi selengkapnya dan untuk mengunduh aplikasi, lihat AWS Console Mobile Application.

Anda dapat masuk ke aplikasi seluler menggunakan kata sandi konsol atau access key Anda. Sebagai praktik terbaik, jangan gunakan access key pengguna root. Sebagai gantinya, kami sangat menyarankan agar selain menggunakan kata sandi atau kunci biometrik di perangkat seluler Anda, Anda membuat pengguna IAM khusus untuk mengelola AWS sumber daya menggunakan aplikasi seluler. Jika perangkat seluler hilang, Anda dapat menghapus akses pengguna IAM.

Untuk masuk menggunakan access key (aplikasi seluler)

1. Buka aplikasi di perangkat seluler Anda.
2. Jika ini pertama kalinya Anda menambahkan identitas ke perangkat, pilih Add an identity (Tambahkan identitas), lalu pilih Access key.

   Jika Anda telah masuk menggunakan identitas lain, pilih ikon menu dan pilih Switch identity (Ganti identitas). Kemudian pilih Sign in as a different identity (Masuk sebagai identitas yang berbeda), lalu Access key.
3. Di halaman Access key, masukkan informasi Anda:

   - Access key ID – Masukkan access key ID Anda.
   - Secret access key – Masukkan secret access key Anda.
   - Nama identitas – Masukkan nama identitas yang akan muncul di aplikasi seluler. Ini tidak harus cocok dengan nama pengguna IAM Anda.
   - PIN Identitas – Buat nomor identifikasi pribadi (PIN) yang akan Anda gunakan untuk masuk di masa mendatang.

   > **Note**
   >
   > Jika Anda mengaktifkan biometrik untuk aplikasi AWS seluler, Anda akan diminta untuk menggunakan sidik jari atau pengenalan wajah untuk verifikasi, bukan PIN. Jika biometrik gagal, Anda mungkin diminta memasukkan PIN.
4. Pilih Verify and add keys (Verifikasi dan tambahkan kunci).

   Sekarang Anda dapat mengakses set sumber daya tertentu menggunakan aplikasi seluler.

Informasi Terkait

Topik berikut memberikan panduan untuk menyiapkan AWS SDK dan penggunaan AWS CLI kunci akses:

- Tetapkan AWS kredenSIAL dan Wilayah di Panduan AWS SDK untuk Java Pengembang
- Menggunakan SDK Store di Panduan Peng AWS SDK untuk .NET embang
- Memberikan KredenSIAL ke SDK di Panduan AWS SDK untuk PHP Pengembang
- Konfigur asi dalam dokumentasi Boto 3 (AWS SDK untuk Python)

- Menggunakan AWS KredenSIAL dalam Panduan AWS Tools for Windows PowerShell Pengguna
- Konfigurasi dan file kreden si dalam Panduan AWS Command Line Interface Pengguna
- Memberikan akses menggunakan peran IAM di Panduan Peng AWS SDK untuk .NET embang
- Konfigurasikan peran IAM untuk Amazon EC2 di AWS SDK for Java 2.x

Menggunakan kunci akses dan kredentif kunci rahasia untuk akses konsol

Dimungkinkan untuk menggunakan kunci akses dan kredentif kunci rahasia untuk Konsol Manajemen AWS akses langsung, bukan hanya AWS CLI. Ini dapat dicapai dengan menggunakan panggilan AWS STS `GetFederationToken` API. Dengan membuat URL konsol menggunakan kredenSIAL sementara dan token yang disediakan oleh`GetFederationToken`, kepala IAM dapat mengakses konsol. Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Perlu diklarifikasi bahwa ketika masuk ke konsol secara langsung menggunakan kredenSIAL pengguna IAM atau root dengan MFA diaktifkan, MFA akan diperlukan. Namun, jika metode yang dijelaskan di atas (menggunakan kredenSIAL sementara dengan`GetFederationToken`) digunakan, MFA TIDAK akan diperlukan.

Mengaudit access key

Anda dapat meninjau kunci AWS akses dalam kode Anda untuk menentukan apakah kunci tersebut berasal dari akun yang Anda miliki. Anda dapat meneruskan ID kunci akses menggunakan `aws sts get-access-key-info` AWS CLI perintah atau operasi `GetAccessKeyInfo` AWS API.

Oper AWS asi AWS CLI dan API mengembalikan ID dari Akun AWS kunci akses milik. ID kunci akses yang dimulai dengan `AKIA` adalah kredenSIAL jangka panjang untuk pengguna IAM atau. Pengguna root akun AWS ID kunci akses yang dimulai dengan `ASIA` adalah kredenSIAL sementara yang dibuat menggunakan AWS STS operasi. Jika akun dalam tanggapan ini adalah milik Anda, Anda dapat masuk sebagai pengguna utama dan meninjau kunci akses pengguna utama Anda. Lalu, Anda dapat menarik laporan kredensial untuk mempelajari pengguna IAM mana yang memilikinya. Untuk mempelajari siapa yang meminta kredentif sementara untuk kunci `ASIA` akses, lihat AWS STS peristiwa di CloudTrail log Anda.

Untuk tujuan keamanan, Anda dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan di AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identias saat mereka

mengasumsikan sebuah peran. Misalnya, Anda dapat meminta pengguna IAM menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS. Untuk informasi selengkapnya, lihat `sts:SourceIdentity`.

Operasi ini tidak menunjukkan status access key. Kuncinya mungkin aktif, tidak aktif, atau dihapus. Kunci aktif mungkin tidak memiliki izin untuk melakukan operasi. Memberikan access key yang dihapus mungkin akan kembali sebagai kesalahan bahwa kunci tidak ada.

AWS Multi-factor otentikasi di IAM

Untuk meningkatkan keamanan, kami sarankan Anda mengonfigurasi otentikasi multi-faktor (MFA) untuk membantu melindungi sumber daya Anda AWS . Anda dapat mengaktifkan MFA untuk semua Akun AWS, termasuk akun mandiri, akun manajemen, dan akun anggota, serta untuk pengguna IAM Anda. Pengguna root akun AWS Sebaiknya gunakan MFA tahan phishing seperti kunci sandi dan kunci keamanan bila memungkinkan. O FIDO-based tentikator ini menggunakan kriptografi kunci publik dan tahan terhadap serangan phishing, man-in-the-middle, dan replay, memberikan tingkat keamanan yang lebih kuat daripada opsi. TOTP-based

MFA diberlakukan untuk semua jenis akun untuk pengguna root mereka. Untuk informasi selengkapnya, lihat Amankan Anda AWS Organizations kredenSIAL pengguna root akun.

Saat Anda mengaktifkan MFA untuk pengguna akar, itu hanya memengaruhi kredensial pengguna akar. Pengguna IAM di akun memiliki identitas berbeda dengan kredensial mereka sendiri, dan setiap identitas memiliki konfigurasi MFA-nya sendiri. Untuk informasi selengkapnya tentang menggunakan MFA untuk melindungi pengguna root, lihatMulti-factor otentikasi untuk Pengguna root akun AWS.

Pengguna Anda Pengguna root akun AWS dan IAM dapat mendaftarkan hingga delapan perangkat MFA jenis apa pun. Mendaftarkan beberapa perangkat MFA dapat memberikan fleksibilitas dan membantu Anda mengurangi risiko gangguan akses jika perangkat hilang atau rusak. Anda hanya perlu satu perangkat MFA untuk masuk ke Konsol Manajemen AWS atau membuat sesi melalui. AWS CLI

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia untuk menggunakan kredenSIAL sementara saat mengakses AWS. Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk secara terpusat mengelola akses ke beberapa Akun AWS dan menyediakan akses masuk MFA-

protected tunggal kepada pengguna ke semua akun yang ditugaskan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di Pusat Identitas IAM atau dengan mudah terhubung ke penyedia identitas yang kompatibel dengan SAML 2.0 yang ada. Untuk informasi selengkapnya, lihat Apa itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

MFA menambahkan keamanan ekstra yang mengharuskan pengguna untuk memberikan otentikasi unik dari mekanisme MFA yang AWS didukung selain kredentif masuk mereka ketika mereka mengakses AWS situs web atau layanan.

Jenis MFA

AWS mendukung jenis MFA berikut:

Daftar Isi

- Kunci sandi dan kunci keamanan
- Aplikasi otentikator virtual
- Token TOTP perangkat keras

Kunci sandi dan kunci keamanan

AWS Identity and Access Management mendukung kunci sandi dan kunci keamanan untuk MFA. Berdasarkan standar FIDO, kunci sandi menggunakan kriptografi kunci publik untuk memberikan otentikasi yang kuat dan tahan phishing yang lebih aman daripada kata sandi. AWS mendukung dua jenis kunci sandi: kunci sandi terikat perangkat (kunci keamanan) dan kunci sandi yang disinkronkan.

- Kunci keamanan: Ini adalah perangkat fisik, seperti YubiKey, digunakan sebagai faktor kedua untuk otentikasi. Satu kunci keamanan dapat mendukung beberapa akun pengguna root dan pengguna IAM.
- Kunci sandi yang disinkron kan: Ini menggunakan manajer kredensia dari penyedia seperti Google, Apple, akun Microsoft, dan layanan pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua.

Anda dapat menggunakan otentikator biometrik bawaan, seperti Touch ID di Apple MacBooks, untuk membuka kunci pengelola kredensi dan masuk. AWS Kunci sandi dibuat dengan penyedia pilihan

Anda menggunakan sidik jari, wajah, atau PIN perangkat Anda. Anda juga dapat menggunakan kunci sandi otentikasi lintas perangkat (CDA) dari satu perangkat, seperti perangkat seluler atau kunci keamanan perangkat keras, untuk masuk di perangkat lain seperti laptop. Untuk informasi selengkapnya, lihat otentikasi lintas perangkat (CDA).

Anda dapat menyinkronkan kunci sandi di seluruh perangkat Anda untuk memfasilitasi login dengan AWS, meningkatkan kegunaan dan pemulihan. Untuk informasi selengkapnya tentang mengaktifkan kunci sandi dan kunci keamanan, lihat. Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)

Aliansi FIDO menyimpan daftar semua produk Bersertifikat FIDO yang kompatibel dengan spesifikasi FIDO.

Aplikasi otentikator virtual

Aplikasi otentikator virtual berjalan di telepon atau perangkat lain dan meniru perangkat fisik. Aplikasi otentikator virtual menerapkan algoritma kata sandi satu kali berbasis waktu (TOTP) dan mendukung beberapa token pada satu perangkat. Pengguna harus mengetikkan kode yang valid dari perangkat saat diminta saat masuk. Setiap token yang diberikan kepada pengguna harus unik. Pengguna tidak dapat mengetikkan kode dari token pengguna lain untuk diautentikasi.

Kami menyarankan Anda menggunakan MFA tahan phishing seperti kunci sandi atau kunci keamanan untuk perlindungan terkuat. Jika Anda belum dapat menggunakan kunci sandi atau kunci keamanan, sebaiknya gunakan perangkat MFA virtual sebagai tindakan sementara sambil menunggu persetujuan pembelian perangkat keras atau saat Anda menunggu perangkat keras tiba. Untuk daftar beberapa aplikasi yang didukung yang dapat Anda gunakan sebagai perangkat MFA virtual, lihat Oten Multi-Factor tikasi (MFA).

Untuk petunjuk tentang cara menyiapkan perangkat MFA virtual untuk pengguna IAM, lihat. Tetapkan perangkat MFA virtual di Konsol Manajemen AWS

> **Note**
>
> Perangkat MFA virtual yang tidak ditetapkan di perangkat Akun AWS Anda akan dihapus saat Anda menambahkan perangkat MFA virtual baru baik melalui Konsol Manajemen AWS atau selama proses masuk. Perangkat MFA virtual yang tidak ditetapkan adalah perangkat di akun Anda tetapi tidak digunakan oleh pengguna root akun atau pengguna IAM untuk proses masuk. Mereka dihapus sehingga perangkat MFA virtual baru dapat ditambahkan ke akun Anda. Ini juga memungkinkan Anda untuk menggunakan kembali nama perangkat.

- Untuk melihat perangkat MFA virtual yang tidak ditetapkan di akun Anda, Anda dapat menggunakan perintah list-virtual- AWS CLI mfa-devices atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_ListVirtualMFADevices.html
- Untuk menonaktifkan perangkat MFA virtual, Anda dapat menggunakan perintah deactivate-mfa- AWS CLI device atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_DeactivateMFADevice.html Perangkat akan menjadi tidak ditugaskan.
- Untuk melampirkan perangkat MFA virtual yang tidak ditetapkan ke pengguna Akun AWS root atau pengguna IAM Anda, Anda memerlukan kode otentikasi yang dihasilkan oleh perangkat bersama dengan perintah enable- AWS CLI mfa-device atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_EnableMFADevice.html

Token TOTP perangkat keras

Perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali berbasis waktu (TOTP). Pengguna harus mengetik kode yang valid dari perangkat di halaman web kedua saat masuk.

Token ini digunakan secara eksklusif dengan Akun AWS. Anda hanya dapat menggunakan token yang memiliki benih token uniknya yang dibagikan dengan aman AWS. Benih token adalah kunci rahasia yang dihasilkan pada saat produksi token. Token yang dibeli dari sumber lain tidak akan berfungsi dengan IAM. Untuk memastikan kompatibilitas, Anda harus membeli perangkat MFA perangkat keras Anda dari salah satu tautan berikut: token OTP atau kartu tampilan OTP.

- Setiap perangkat MFA yang ditetapkan ke pengguna harus unik. Pengguna tidak dapat mengetik kode dari perangkat pengguna lain untuk diautentikasi. Untuk informasi tentang perangkat MFA perangkat keras yang didukung, lihat Oten Multi-Factor tikasi (MFA).
- Jika Anda ingin menggunakan perangkat MFA fisik, sebaiknya gunakan kunci keamanan sebagai alternatif perangkat TOTP perangkat keras. Kunci keamanan tidak memerlukan baterai, tahan phishing, dan mendukung banyak pengguna pada satu perangkat.

Anda dapat mengaktifkan kunci sandi atau kunci keamanan dari Konsol Manajemen AWS satu-satunya, bukan dari API AWS CLI or AWS . Sebelum Anda dapat mengaktifkan kunci keamanan, Anda harus memiliki akses fisik ke perangkat.

Untuk petunjuk tentang cara menyiapkan token TOTP perangkat keras untuk pengguna IAM, lihat. Tetapkan token TOTP perangkat keras di Konsol Manajemen AWS

> **Note**
>
> MFA berbasis pesan teks SMS — meng AWS akhiri dukungan untuk mengaktifkan otentikasi multi-faktor SMS (MFA). Kami menyarankan pelanggan yang memiliki pengguna IAM yang menggunakan MFA berbasis pesan teks SMS beralih ke salah satu metode alternatif berikut: Kunci sandi atau kunci keamanan, perangkat M FA virtual (berbasis perangkat lunak), atau perangkat MFA perangkat keras. ??? Anda dapat mengidentifikasi pengguna di akun Anda dengan perangkat SMS MFA yang ditetapkan. Di konsol IAM, pilih Pengguna dari panel navigasi, dan cari pengguna dengan SMS dalam kolom tabel MFA.

Rekomendasi MFA

Untuk membantu mengamankan AWS identitas Anda, ikuti rekomendasi ini untuk otentikasi MFA.

- Sebaiknya gunakan MFA tahan phishing, seperti kunci sandi dan kunci keamanan, sebagai perangkat MFA Anda. O FIDO-based tentikator ini memberikan perlindungan terkuat terhadap serangan seperti phishing.
- Sebaiknya aktifkan beberapa perangkat MFA untuk pengguna Pengguna root akun AWS dan IAM di perangkat Anda. Akun AWS Ini memungkinkan Anda untuk menaikkan bilah keamanan di Anda Akun AWS dan menyederhanakan pengelolaan akses ke pengguna yang sangat istimewa, seperti Pengguna root akun AWS.
- Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Anda Pengguna root akun AWS dan IAM. Dengan beberapa perangkat MFA, Anda hanya perlu satu perangkat MFA untuk masuk ke Konsol Manajemen AWS atau membuat sesi melalui AWS CLI sebagai pengguna tersebut. Pengguna IAM harus melakukan otentikasi dengan perangkat MFA yang ada untuk mengaktifkan atau menonaktifkan perangkat MFA tambahan.
- Jika perangkat MFA hilang, dicuri, atau tidak dapat diakses, Anda dapat menggunakan salah satu perangkat MFA yang tersisa untuk mengakses Akun AWS tanpa melakukan prosedur Akun AWS pemulihan. Jika perangkat MFA hilang atau dicuri, perangkat tersebut harus dipisahkan dari prinsipal IAM yang terkait dengannya.
- Penggunaan beberapa MFA memungkinkan karyawan Anda di lokasi yang tersebar secara geografis atau bekerja dari jarak jauh untuk menggunakan MFA berbasis perangkat keras untuk

mengakses AWS tanpa harus mengoordinasikan pertukaran fisik perangkat keras tunggal antar karyawan.

- Penggunaan perangkat MFA tambahan untuk prinsipal IAM memungkinkan Anda menggunakan satu atau lebih MFA untuk penggunaan sehari-hari, sambil juga menjaga perangkat MFA fisik di lokasi fisik yang aman seperti brankas atau aman untuk cadangan dan redundansi.

> **Catatan**
>
> - Anda tidak dapat meneruskan informasi MFA untuk kunci keamanan atau kunci sandi ke operasi AWS STS API untuk meminta kredenSIAL sementara. Anda dapat memperoleh kredenSIAL untuk digunakan dengan AWS SDK AWS CLI dan saat menggunakan kunci keamanan atau kunci sandi dengan menjalankan perintah. `aws login`
> - Anda tidak dapat menggunakan AWS CLI perintah atau operasi AWS API untuk mengaktifkan kunci keamanan FIDO.
> - Anda tidak dapat menggunakan nama yang sama untuk lebih dari satu pengguna root atau perangkat IAM MFA.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang MFA.

- Untuk informasi selengkapnya tentang menggunakan MFA untuk mengakses AWS, lihatMFA mengaktifkan login.
- Anda dapat menggunakan Pusat Identitas IAM untuk mengaktifkan akses MFA yang aman ke portal AWS akses Anda, aplikasi terintegrasi IAM Identity Center, dan. AWS CLI Untuk informasi selengkapnya, lihat Meng aktifkan MFA di Pusat Identitas IAM.
- Untuk memverifikasi apakah MFA sudah diaktifkan untuk pengguna Anda, lihatPeriksa status MFA.

Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS

Kunci sandi adalah jenis perangkat otentikasi multi-faktor (MFA) yang dapat Anda gunakan untuk melindungi sumber daya Anda. AWS AWS mendukung kunci sandi yang disinkronkan dan kunci sandi terikat perangkat yang juga dikenal sebagai kunci keamanan.

Kunci sandi yang disinkronkan memungkinkan pengguna IAM mengakses kredenSIAL masuk FIDO mereka di banyak perangkat mereka, bahkan yang baru, tanpa harus mendaftarkan ulang setiap perangkat di setiap akun. Kunci sandi yang disinkronkan mencakup manajer kredensia pihak pertama seperti Google, Apple, dan Microsoft dan manajer kredensia pihak ketiga seperti 1Password, Dashlane, dan Bitwarden sebagai faktor kedua. Anda juga dapat menggunakan biometrik pada perangkat (misalnya, TouchID, FaceID) untuk membuka kunci pengelola kredensia pilihan Anda untuk menggunakan kunci sandi.

Atau, kunci sandi yang terikat perangkat terikat ke kunci keamanan FIDO yang Anda colokkan ke port USB di komputer Anda dan kemudian ketuk saat diminta untuk menyelesaikan proses masuk dengan aman. Jika Anda sudah menggunakan kunci keamanan FIDO dengan layanan lain, dan memiliki konfigurasi yang AWS didukung (misalnya, Seri YubiKey 5 dari Yubico), Anda juga dapat menggunakannya dengan. AWS Jika tidak, Anda perlu membeli kunci keamanan FIDO jika Anda ingin menggunakan WebAuthn untuk MFA di. AWS Selain itu, kunci keamanan FIDO dapat mendukung beberapa pengguna IAM atau root pada perangkat yang sama, meningkatkan utilitas mereka untuk keamanan akun. Untuk spesifikasi dan informasi pembelian untuk kedua jenis perangkat, lihat Oten Multi-Factor tikasi.

Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Anda Pengguna root akun AWS dan IAM. Dengan beberapa perangkat MFA, Anda hanya perlu satu perangkat MFA untuk masuk ke Konsol Manajemen AWS atau membuat sesi melalui AWS CLI sebagai pengguna tersebut. Kami menyarankan Anda mendaftarkan beberapa perangkat MFA. Misalnya, Anda dapat mendaftarkan autentikator bawaan dan juga mendaftarkan kunci keamanan yang Anda simpan di lokasi yang aman secara fisik. Jika Anda tidak dapat menggunakan autentikator bawaan, maka Anda dapat menggunakan kunci keamanan terdaftar Anda. Untuk aplikasi otentikator, kami juga menyarankan untuk mengaktifkan fitur pencadangan atau sinkronisasi cloud di aplikasi tersebut untuk membantu Anda menghindari kehilangan akses ke akun jika Anda kehilangan atau merusak perangkat Anda dengan aplikasi otentikator.

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia untuk menggunakan kredenSIAL sementara saat mengakses AWS. Pengguna Anda dapat bergabung AWS dengan penyedia identitas tempat mereka melakukan otentikasi dengan kredentif perusahaan dan konfigurasi MFA mereka. Untuk mengelola akses ke AWS dan aplikasi bisnis, sebaiknya gunakan Pusat Identitas IAM. Untuk informasi selengkapnya, lihat Panduan Pengguna Pusat Identitas IAM.

Topik

- Izin diperlukan
- Mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM Anda sendiri (konsol)
- Mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM lain (konsol)
- Mengganti kunci sandi atau kunci keamanan
- Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Izin diperlukan

Untuk mengelola kunci sandi FIDO untuk pengguna IAM Anda sendiri sambil melindungi MFA-related tindakan sensitif, Anda harus memiliki izin dari kebijakan berikut:

> **Note**
>
> Nilai ARN adalah nilai statis dan bukan merupakan indikator protokol apa yang digunakan untuk mendaftarkan otentikator. Kami telah menghentikan U2F, jadi semua implementasi baru digunakan. WebAuthn

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
```

```
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM Anda sendiri (konsol)

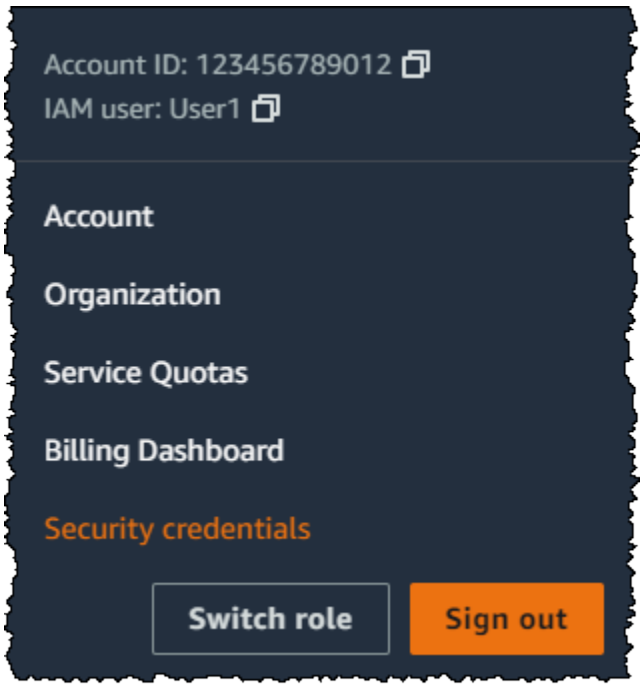
Anda dapat mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM Anda sendiri dari Konsol Manajemen AWS satu-satunya, bukan dari atau API. AWS CLI AWS Sebelum Anda dapat mengaktifkan kunci keamanan, Anda harus memiliki akses fisik ke perangkat.

Untuk mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun IAM Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun untuk dialihkan ke halaman masuk pengguna IAM untuk akun Anda.

   Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi bilitas keamanan.



3. Pada halaman pengguna IAM yang dipilih, pilih tab Kredensi bilitas keamanan.
4. Di bawah Multi-factor otentikasi (MFA), pilih Tet apkan perangkat MFA.
5. Pada halaman nama perangkat MFA, masukkan nama Perangkat, pilih Kunci sandi atau Kunci Keamanan, lalu pilih Berikutnya.
6. Di Siapkan perangkat, atur kunci sandi Anda. Buat kunci sandi dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci keamanan FIDO ke port USB komputer Anda dan mengetuknya.
7. Ikuti petunjuk pada browser Anda, lalu pilih L anjutkan.

Anda sekarang telah mendaftarkan kunci sandi atau kunci keamanan Anda untuk digunakan bersama. AWS Untuk informasi tentang menggunakan MFA dengan Konsol Manajemen AWS, lihatMFA mengaktifkan login.

Mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM lain (konsol)

Anda dapat mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM lain dari Konsol Manajemen AWS satu-satunya, bukan dari API or. AWS CLI AWS

Untuk mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM lain (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Users (Pengguna).
3. Di bawah Pengguna, pilih nama pengguna yang ingin Anda aktifkan MFA.
4. Pada halaman pengguna IAM yang dipilih, pilih tab Security Credentials.
5. Di bawah Multi-factor otentikasi (MFA), pilih Tet apkan perangkat MFA.
6. Pada halaman nama perangkat MFA, masukkan nama Perangkat, pilih Kunci sandi atau Kunci Keamanan, lalu pilih Berikutnya.
7. Di Siapkan perangkat, atur kunci sandi Anda. Buat kunci sandi dengan data biometrik seperti wajah atau sidik jari Anda, dengan pin perangkat, atau dengan memasukkan kunci keamanan FIDO ke port USB komputer Anda dan mengetuknya.
8. Ikuti petunjuk pada browser Anda, lalu pilih L anjutkan.

Anda sekarang telah mendaftarkan kunci sandi atau kunci keamanan untuk digunakan pengguna IAM lain. AWS Untuk informasi tentang menggunakan MFA dengan Konsol Manajemen AWS, lihatMFA mengaktifkan login.

Mengganti kunci sandi atau kunci keamanan

Anda dapat memiliki hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung yang ditetapkan ke pengguna pada satu waktu dengan pengguna Anda Pengguna root akun AWS dan IAM. Jika pengguna kehilangan autentikator FIDO atau perlu menggantinya karena alasan apa pun, Anda harus terlebih dahulu menonaktifkan otentikator FIDO lama. Kemudian Anda dapat menambahkan perangkat MFA baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan pengguna IAM, lihatNonaktifkan perangkat MFA.
- Untuk menambahkan kunci keamanan FIDO baru untuk pengguna IAM, lihat. Mengaktifkan kunci sandi atau kunci keamanan untuk pengguna IAM Anda sendiri (konsol)

Jika Anda tidak memiliki akses ke kunci sandi atau kunci keamanan baru, Anda dapat mengaktifkan perangkat MFA virtual atau token TOTP perangkat keras baru. Lihat salah satu dari yang berikut untuk petunjuk:

- Tetapkan perangkat MFA virtual di Konsol Manajemen AWS
- Tetapkan token TOTP perangkat keras di Konsol Manajemen AWS

Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan

Anda dapat menggunakan kunci sandi terikat perangkat FIDO2, juga dikenal sebagai kunci keamanan, sebagai metode otentikasi multi-faktor (MFA) dengan IAM menggunakan konfigurasi yang saat ini didukung. Ini termasuk perangkat FIDO2 yang didukung oleh IAM dan browser yang mendukung FIDO2.

Sebelum mendaftarkan perangkat FIDO2 Anda, periksa apakah Anda menggunakan versi browser dan sistem operasi (OS) terbaru. Fitur mungkin berperilaku berbeda di berbagai browser, otentikator, dan klien OS. Jika pendaftaran perangkat Anda gagal pada satu browser, Anda dapat mencoba mendaftar dengan browser lain.

FIDO2 adalah standar otentikasi terbuka dan perpanjangan dari FIDO U2F, menawarkan tingkat keamanan tinggi yang sama berdasarkan kriptografi kunci publik. FIDO2 terdiri dari spesifikasi W3C Web Authentication (WebAuthn API) dan FIDO Alliance Client-to-Authenticator Protocol (CTAP), sebuah protokol lapisan aplikasi. CTAP memungkinkan komunikasi antara klien atau platform, seperti browser atau sistem operasi, dengan otentikator eksternal.

Saat Anda mengaktifkan autentikator Bersertifikat FIDO di AWS, kunci keamanan membuat pasangan kunci baru untuk digunakan hanya. AWS Pertama, Anda memasukkan kredensial Anda. Saat diminta, Anda mengetuk kunci keamanan, yang menanggapi tantangan otentikasi yang dikeluarkan oleh AWS. Untuk mempelajari lebih lanjut tentang standar FIDO2, lihat Proyek FIDO2.

Perangkat FIDO2 didukung oleh AWS

IAM mendukung perangkat keamanan FIDO2 yang terhubung ke perangkat Anda melalui USB,Bluetooth, atau NFC. IAM juga mendukung otentikator platform seperti TouchID atau FaceID. IAM tidak mendukung pendaftaran kunci sandi lokal untuk Windows Hello. Untuk membuat dan menggunakan kunci sandi, pengguna Windows harus menggunakan otentikasi lintas perangkat di mana Anda menggunakan kunci sandi dari satu perangkat seperti perangkat seluler atau kunci keamanan perangkat keras untuk masuk di perangkat lain seperti laptop.

> **Note**
>
> AWS memerlukan akses ke port USB fisik di komputer Anda untuk memverifikasi perangkat FIDO2 Anda. Kunci keamanan tidak akan berfungsi dengan mesin virtual, koneksi jarak jauh, atau mode penyamaran browser.

Aliansi FIDO menyimpan daftar semua produk FIDO2 yang kompatibel dengan spesifikasi FIDO.

Browser yang mendukung FIDO2

Ketersediaan perangkat keamanan FIDO2 yang berjalan di browser web tergantung pada kombinasi browser dan sistem operasi. Browser berikut saat ini mendukung penggunaan kunci keamanan:

Peramban web	MacOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Ya	Ya	Ya	Ya	Tidak
Safari	Ya	Tidak	Tidak	Ya	Tidak
Edge	Ya	Ya	Tidak	Ya	Tidak
Firefox	Ya	Ya	Tidak	Ya	Tidak

> **Note**
>
> Sebagian besar versi Firefox yang saat ini mendukung FIDO2 tidak mengaktifkan dukungan secara default. Untuk petunjuk tentang mengaktifkan dukungan FIDO2 di Firefox, lihat. Memecahkan Masalah Kunci Sandi dan Kunci Keamanan FIDO
> Firefox di macOS mungkin tidak sepenuhnya mendukung alur kerja otentikasi lintas perangkat untuk kunci sandi. Anda mungkin mendapatkan perintah untuk menyentuh kunci keamanan alih-alih melanjutkan dengan otentikasi lintas perangkat. Sebaiknya gunakan browser yang berbeda, seperti Chrome atau Safari, untuk masuk dengan kunci sandi di macOS.

Untuk informasi selengkapnya tentang dukungan browser untuk FIDO2-Certified perangkat seperti YubiKey, lihat Sistem operasi dan dukungan browser web untuk FIDO2 dan U2F.

Plugin peramban

AWS hanya mendukung browser yang secara native mendukung FIDO2. AWS tidak mendukung penggunaan plugin untuk menambahkan dukungan browser FIDO2. Beberapa plugin browser tidak kompatibel dengan standar FIDO2 dan dapat menyebabkan hasil yang tidak terduga dengan kunci keamanan FIDO2.

Untuk informasi tentang menonaktifkan plugin peramban dan tips pemecahan masalah lainnya, lihat Saya tidak dapat mengaktifkan kunci keamanan FIDO saya.

Sertifikasi perangkat

Kami menangkap dan menetapkan sertifikasi terkait perangkat, seperti validasi FIPS dan tingkat sertifikasi FIDO, hanya selama pendaftaran kunci keamanan. Sertifikasi perangkat Anda diambil dari Layanan Metadata Aliansi FIDO (MDS).

Jika status sertifikasi atau tingkat kunci keamanan Anda berubah, itu tidak akan tercermin dalam tag perangkat secara otomatis. Untuk memperbarui informasi sertifikasi perangkat, daftarkan perangkat lagi untuk mengambil informasi sertifikasi yang diperbarui.

AWS menyediakan jenis sertifikasi berikut sebagai kunci kondisi selama pendaftaran perangkat, diperoleh dari tingkat sertifikasi FIDO MDS: FIPS-140-2, FIPS-140-3, dan FIDO. Anda dapat membatasi pendaftaran otentikator dalam kebijakan IAM berdasarkan jenis dan tingkat sertifikasi. Untuk informasi lebih lanjut, lihat kebijakan di bawah ini.

Contoh kebijakan untuk sertifikasi perangkat

Kasus penggunaan berikut menunjukkan contoh kebijakan yang memungkinkan Anda mendaftarkan perangkat MFA dengan sertifikasi FIPS.

Topik

- Kasus penggunaan 1: Izinkan pendaftaran hanya perangkat yang memiliki sertifikasi FIPS-140-2 L2
- Kasus penggunaan 2: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 dan FIDO L1
- Kasus penggunaan 3: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 atau FIPS-140-3 L2
- Kasus penggunaan 4: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 dan mendukung jenis MFA lainnya seperti otentikator virtual dan TOTP perangkat keras

Kasus penggunaan 1: Izinkan pendaftaran hanya perangkat yang memiliki sertifikasi FIPS-140-2 L2

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Kasus penggunaan 2: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 dan FIDO L1

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

Kasus penggunaan 3: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 atau FIPS-140-3 L2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

Kasus penggunaan 4: Izinkan mendaftarkan perangkat yang memiliki sertifikasi FIPS-140-2 L2 dan mendukung jenis MFA lainnya seperti otentikator virtual dan TOTP perangkat keras

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Create"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
```

```
          "iam:RegisterSecurityKey": "Activate",
          "iam:FIDO-FIPS-140-2-certification": "L2"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "Null": {
          "iam:RegisterSecurityKey": "true"
        }
      }
    }
  ]
}
```

AWS CLI and AWS API

AWS mendukung penggunaan kunci sandi dan kunci keamanan hanya di. Konsol Manajemen AWS Menggunakan kunci sandi dan kunci keamanan untuk MFA tidak didukung di AWS API AWS CLI dan, atau untuk akses ke MFA-protected operasi API.

Sumber daya tambahan

- Untuk informasi selengkapnya tentang menggunakan kunci sandi dan kunci keamanan AWS, lihat. Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS
- Untuk bantuan pemecahan masalah kunci sandi dan kunci keamanan di AWS, lihat. Memecahkan Masalah Kunci Sandi dan Kunci Keamanan FIDO
- Untuk informasi industri umum tentang dukungan FIDO2, lihat Proyek FIDO2.

Tetapkan perangkat MFA virtual di Konsol Manajemen AWS

> **Important**
>
> AWS merekomendasikan agar Anda menggunakan kunci sandi atau kunci keamanan untuk MFA AWS, jika memungkinkan. Untuk informasi selengkapnya, lihat Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS.

Anda dapat menggunakan ponsel atau perangkat lain sebagai perangkat Autentikasi Multi-Faktor (MFA). Untuk melakukannya, pasang aplikasi seluler yang sesuai dengan RFC 6238, algoritma TOTP berbasis-standar (kata sandi satu kali berbasis waktu). Aplikasi ini menghasilkan kode otentikasi enam digit. Karena otentikator dapat berjalan pada perangkat seluler yang tidak aman, dan kode berpotensi dibagikan dengan pihak yang tidak berwenang, TOTP-based MFA tidak memberikan tingkat keamanan yang sama dengan opsi tahan phishing seperti kunci keamanan FIDO2 dan kunci sandi. Sebaiknya gunakan kunci sandi atau kunci keamanan untuk MFA untuk perlindungan terkuat terhadap serangan seperti phishing.

Jika Anda belum dapat menggunakan kunci sandi atau kunci keamanan, sebaiknya gunakan perangkat MFA virtual sebagai tindakan sementara sambil menunggu persetujuan pembelian perangkat keras atau perangkat keras Anda tiba.

Sebagian besar aplikasi MFA virtual mendukung pembuatan beberapa perangkat virtual, memungkinkan Anda menggunakan aplikasi yang sama untuk beberapa Akun AWS atau pengguna. Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA apa pun dengan pengguna Anda Pengguna root akun AWS dan IAM. Anda hanya perlu satu perangkat MFA untuk masuk ke Konsol Manajemen AWS atau membuat sesi melalui. AWS CLI Kami menyarankan Anda mendaftarkan beberapa perangkat MFA. Untuk aplikasi otentikator, kami juga menyarankan untuk mengaktifkan fitur pencadangan atau sinkronisasi cloud untuk membantu Anda menghindari kehilangan akses ke akun jika perangkat Anda hilang atau rusak.

AWS membutuhkan aplikasi MFA virtual yang menghasilkan OTP enam digit. Untuk daftar aplikasi MFA virtual yang dapat Anda gunakan, lihat Oten Multi-Factor tik asi.

Topik

- Izin diperlukan
- Aktifkan perangkat MFA virtual untuk pengguna IAM (konsol)
- Mengganti perangkat MFA virtual

Izin diperlukan

Untuk mengelola perangkat MFA virtual untuk pengguna IAM Anda, Anda harus memiliki izin dari kebijakan berikut: AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan.

Aktifkan perangkat MFA virtual untuk pengguna IAM (konsol)

Anda dapat menggunakan IAM di Konsol Manajemen AWS untuk mengaktifkan dan mengelola perangkat MFA virtual untuk pengguna IAM di akun Anda. Anda dapat melampirkan tanda ke sumber daya IAM, termasuk perangkat MFA virtual, untuk mengidentifikasi, mengatur, dan mengontrol akses ke perangkat tersebut. Anda dapat menandai perangkat MFA virtual hanya saat Anda menggunakan AWS API AWS CLI atau. Untuk mengaktifkan dan mengelola perangkat MFA menggunakan AWS API AWS CLI atau, lihatTetapkan perangkat MFA di AWS CLI atau API AWS. Untuk informasi selengkapnya tentang menandai sumber daya IAM, lihat. Tag untuk AWS Identity and Access Management sumber daya

> **Note**
>
> Anda harus memiliki akses fisik ke perangkat keras yang akan menjadi host perangkat MFA virtual pengguna untuk mengonfigurasi MFA. Misalnya, Anda dapat mengonfigurasi MFA untuk pengguna yang akan menggunakan perangkat MFA virtual yang berjalan di smartphone. Dalam hal ini, Anda harus memiliki smartphone yang tersedia untuk menyelesaikan wizard. Oleh karena itu, Anda mungkin ingin membiarkan pengguna mengonfigurasi dan mengelola perangkat MFA virtual mereka sendiri. Dalam kasus tersebut, Anda harus memberikan izin kepada pengguna untuk melakukan tindakan IAM yang diperlukan. Untuk informasi selengkapnya dan untuk contoh kebijakan IAM yang memberikan izin ini, lihat kebijakan Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan dan contoh.

Untuk mengaktifkan perangkat MFA virtual untuk pengguna IAM (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Dalam daftar Pengguna, pilih nama pengguna IAM.
4. Pilih tab Kredensial Keamanan. Di bawah Multi-factor otentikasi (MFA), pilih Tet apkan perangkat MFA.
5. Di wizard, ketik nama Perangkat, pilih aplikasi Authenticator, lalu pilih Next.

IAM menghasilkan dan menampilkan informasi konfigurasi untuk perangkat MFA virtual, termasuk grafik kode QR. Grafik adalah representasi "kunci konfigurasi rahasia" yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.

6. Buka aplikasi MFA virtual Anda. Untuk daftar aplikasi yang dapat Anda gunakan untuk menghosting perangkat MFA virtual, lihat Oten Multi-Factor tik asi.

   Jika aplikasi MFA virtual mendukung beberapa perangkat atau akun MFA virtual, pilih opsi untuk membuat perangkat atau akun MFA virtual baru.

7. Tentukan apakah aplikasi MFA mendukung kode QR, kemudian lakukan salah satu hal berikut:

   - Dari wizard, pilih Tampilkan kode QR, lalu gunakan aplikasi untuk memindai kode QR. Ini mungkin ikon kamera atau opsi Kode Pindai yang menggunakan kamera perangkat untuk memindai kode.
   - Dari wizard, pilih Tampilkan kunci rahasia, lalu ketik kunci rahasia ke dalam aplikasi MFA Anda.

   Saat Anda selesai, perangkat MFA virtual mulai membuat kata sandi sekali pakai.

8. Pada halaman Siapkan perangkat, di kotak kode MFA 1, ketik kata sandi satu kali yang saat ini muncul di perangkat MFA virtual. Tunggu hingga 30 detik untuk membuat kata sandi sekali pakai yang baru. Lalu ketikkan kata sandi satu kali kedua ke kotak Kode MFA 2. Pilih Tambahkan MFA.

   > **Important**
   >
   > Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat MFA virtual sekarang siap digunakan dengan AWS. Untuk informasi tentang menggunakan MFA dengan Konsol Manajemen AWS, lihatMFA mengaktifkan login.

> **Note**
>
> Perangkat MFA virtual yang tidak ditetapkan di perangkat Akun AWS Anda akan dihapus saat Anda menambahkan perangkat MFA virtual baru baik melalui Konsol Manajemen AWS atau selama proses masuk. Perangkat MFA virtual yang tidak ditetapkan adalah perangkat di akun Anda tetapi tidak digunakan oleh pengguna root akun atau pengguna IAM untuk proses masuk. Mereka dihapus sehingga perangkat MFA virtual baru dapat ditambahkan ke akun Anda. Ini juga memungkinkan Anda untuk menggunakan kembali nama perangkat.
>
> - Untuk melihat perangkat MFA virtual yang tidak ditetapkan di akun Anda, Anda dapat menggunakan perintah list-virtual- AWS CLI mfa-devices atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_ListVirtualMFADevices.html
> - Untuk menonaktifkan perangkat MFA virtual, Anda dapat menggunakan perintah deactivate-mfa- AWS CLI device atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_DeactivateMFADevice.html Perangkat akan menjadi tidak ditugaskan.
> - Untuk melampirkan perangkat MFA virtual yang tidak ditetapkan ke pengguna Akun AWS root atau pengguna IAM Anda, Anda memerlukan kode otentikasi yang dihasilkan oleh perangkat bersama dengan perintah enable- AWS CLI mfa-device atau panggilan API. https://docs.aws.amazon.com/IAM/latest/APIReference/API_EnableMFADevice.html

Mengganti perangkat MFA virtual

Pengguna Anda Pengguna root akun AWS dan IAM dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA apa pun. Jika pengguna kehilangan perangkat atau perlu menggantinya karena alasan apa pun, nonaktifkan perangkat lama. Kemudian Anda dapat menambahkan perangkat baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan pengguna IAM lain, lihat Nonaktifkan perangkat MFA.
- Untuk menambahkan perangkat MFA virtual pengganti bagi pengguna IAM lain, ikuti langkah-langkah dalam prosedur Aktifkan perangkat MFA virtual untuk pengguna IAM (konsol) di atas.
- Untuk menambahkan perangkat MFA virtual pengganti untuk Pengguna root akun AWS, ikuti langkah-langkah dalam prosedurAktifkan perangkat MFA virtual untuk pengguna root (konsol).

Tetapkan token TOTP perangkat keras di Konsol Manajemen AWS

> **Important**
>
> AWS merekomendasikan agar Anda menggunakan kunci sandi atau kunci keamanan untuk MFA AWS, jika memungkinkan. Untuk informasi selengkapnya, lihat Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS.

Token TOTP perangkat keras menghasilkan kode numerik enam digit berdasarkan algoritma kata sandi satu kali (TOTP) berbasis waktu. Pengguna harus memasukkan kode yang valid dari perangkat saat diminta selama proses masuk. Setiap perangkat MFA yang ditetapkan ke pengguna harus unik; pengguna tidak dapat mengetikkan kode dari perangkat pengguna lain untuk diautentikasi. Perangkat MFA tidak dapat dibagikan di seluruh akun atau pengguna.

Token TOTP perangkat keras dan kunci keamanan FIDO keduanya merupakan perangkat fisik yang Anda beli. Perangkat MFA perangkat keras menghasilkan kode TOTP untuk otentikasi saat Anda masuk. AWS Mereka bergantung pada baterai, yang mungkin perlu diganti dan disinkronisasi ulang seiring AWS waktu. Kunci keamanan FIDO, yang menggunakan kriptografi kunci publik, tidak memerlukan baterai dan menawarkan proses otentikasi yang mulus. Sebaiknya gunakan kunci keamanan FIDO untuk resistensi phishing mereka, yang menyediakan alternatif yang lebih aman untuk perangkat TOTP. Selain itu, kunci keamanan FIDO dapat mendukung beberapa pengguna IAM atau root pada perangkat yang sama, meningkatkan utilitas mereka untuk keamanan akun. Untuk spesifikasi dan informasi pembelian untuk kedua jenis perangkat, lihat Oten Multi-Factor tikasi.

Anda dapat mengaktifkan token TOTP perangkat keras untuk pengguna IAM dari Konsol Manajemen AWS, baris perintah, atau IAM API. Untuk mengaktifkan perangkat MFA untuk Anda Pengguna root akun AWS, lihatAktifkan token TOTP perangkat keras untuk pengguna root (konsol).

Anda dapat mendaftarkan hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dengan pengguna Anda Pengguna root akun AWS dan IAM. Dengan beberapa perangkat MFA, Anda hanya perlu satu perangkat MFA untuk masuk ke Konsol Manajemen AWS atau membuat sesi melalui AWS CLI sebagai pengguna tersebut.

> **Important**
>
> Sebaiknya aktifkan beberapa perangkat MFA bagi pengguna untuk akses berkelanjutan ke akun Anda jika perangkat MFA hilang atau tidak dapat diakses.

> **Note**
>
> Jika Anda ingin mengaktifkan perangkat MFA dari baris perintah, gunakan `aws iam enable-mfa-device`. Untuk mengaktifkan perangkat MFA dengan IAM API, gunakan operasi. `EnableMFADevice`

Topik

- Izin diperlukan
- Aktifkan token TOTP perangkat keras untuk pengguna IAM Anda sendiri (konsol)
- Mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol)
- Mengganti perangkat MFA fisik

Izin diperlukan

Untuk mengelola token TOTP perangkat keras untuk pengguna IAM Anda sendiri sambil melindungi MFA-related tindakan sensitif, Anda harus memiliki izin dari kebijakan berikut:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
```

```
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Aktifkan token TOTP perangkat keras untuk pengguna IAM Anda sendiri (konsol)

Anda dapat mengaktifkan token TOTP perangkat keras Anda sendiri dari Konsol Manajemen AWS.

> **Note**
>
> Sebelum Anda dapat mengaktifkan token TOTP perangkat keras, Anda harus memiliki akses fisik ke perangkat.

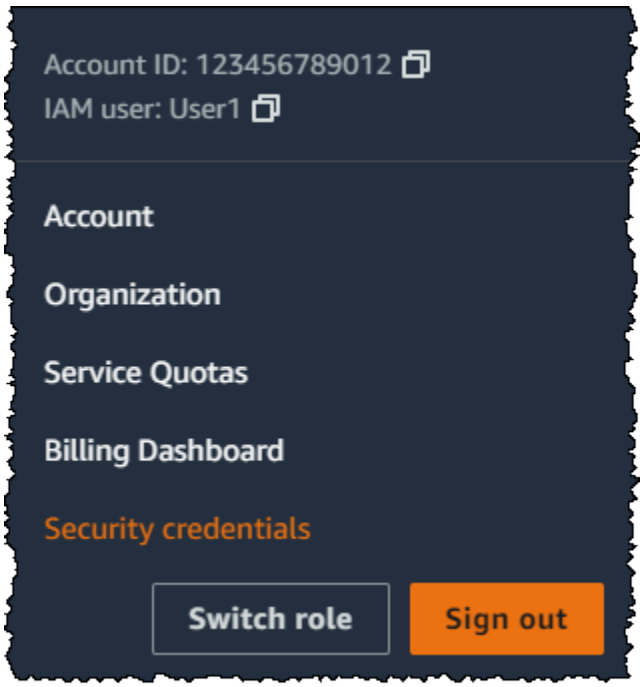
Untuk mengaktifkan token TOTP perangkat keras untuk pengguna IAM Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

   > **Note**
   >
   > Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun IAM Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun untuk dialihkan ke halaman masuk pengguna IAM untuk akun Anda.

Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi bilitas keamanan.



3. Pada tab kreden AWS SIAL IAM, di bagian Multi-factor otentikasi (MFA), pilih Tet apkan perangkat MFA.
4. Di wizard, ketik nama Perangkat, pilih token TOTP Perangkat Keras, lalu pilih Ber ikutnya.
5. Ketik nomor seri perangkat. Nomor seri biasanya ada di bagian belakang perangkat.
6. Di kotak Kode MFA 1, ketik angka enam-digit yang ditampilkan oleh perangkat MFA. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.



7. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke kotak Kode MFA 2. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.
8. Pilih Tambahkan MFA.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak akan sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi lebih lanjut tentang penggunaan MFA dengan Konsol Manajemen AWS, lihat MFA mengaktifkan login.

Mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol)

Anda dapat mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain dari. Konsol Manajemen AWS

Untuk mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Users (Pengguna).
3. Pilih nama pengguna yang ingin Anda aktifkan MFA.
4. Pilih tab Kredensial Keamanan. Di bawah Multi-factor otentikasi (MFA), pilih Tet apkan perangkat MFA.
5. Di wizard, ketik nama Perangkat, pilih token TOTP Perangkat Keras, lalu pilih Ber ikutnya.
6. Ketik nomor seri perangkat. Nomor seri biasanya ada di bagian belakang perangkat.
7. Di kotak Kode MFA 1, ketik angka enam-digit yang ditampilkan oleh perangkat MFA. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor.



8. Tunggu 30 detik saat perangkat menyegarkan kode, lalu ketik angka enam digit berikutnya ke kotak Kode MFA 2. Anda mungkin perlu menekan tombol di bagian depan perangkat untuk menampilkan nomor kedua.

9. Pilih Tambahkan MFA.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak akan sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

Perangkat siap digunakan dengan AWS. Untuk informasi lebih lanjut tentang penggunaan MFA dengan Konsol Manajemen AWS, lihat MFA mengaktifkan login.

Mengganti perangkat MFA fisik

Anda dapat memiliki hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung yang ditetapkan ke pengguna pada satu waktu dengan pengguna Anda Pengguna root akun AWS dan IAM. Jika pengguna kehilangan perangkat atau perlu mengganti karena alasan apa pun, Anda harus menonaktifkan perangkat lama terlebih dahulu. Kemudian Anda dapat menambahkan perangkat baru untuk pengguna.

- Untuk menonaktifkan perangkat yang saat ini terkait dengan pengguna, lihat Nonaktifkan perangkat MFA.
- Untuk menambahkan token TOTP perangkat keras pengganti untuk pengguna IAM, ikuti langkah-langkah dalam prosedur Mengaktifkan token TOTP perangkat keras untuk pengguna IAM lain (konsol) sebelumnya dalam topik ini.
- Untuk menambahkan token TOTP perangkat keras pengganti untuk Pengguna root akun AWS, ikuti langkah-langkah dalam prosedur Aktifkan token TOTP perangkat keras untuk pengguna root (konsol) sebelumnya dalam topik ini.

Tetapkan perangkat MFA di AWS CLI atau API AWS

Anda dapat menggunakan AWS CLI perintah atau operasi AWS API untuk mengaktifkan perangkat MFA virtual untuk pengguna IAM. Anda tidak dapat mengaktifkan perangkat MFA untuk Pengguna root akun AWS dengan AWS CLI, AWS API, Alat untuk Windows PowerShell, atau alat baris perintah lainnya. Namun, Anda dapat menggunakan Konsol Manajemen AWS untuk mengaktifkan perangkat MFA untuk pengguna root.

Saat Anda mengaktifkan perangkat MFA dari Konsol Manajemen AWS, konsol melakukan beberapa langkah untuk Anda. Jika Anda malah membuat perangkat virtual menggunakan AWS CLI, Alat untuk Windows PowerShell, atau AWS API, maka Anda harus melakukan langkah-langkah secara manual dan dalam urutan yang benar. Misalnya, untuk membuat perangkat MFA virtual, Anda harus membuat objek IAM dan mengekstrak kode sebagai sebuah string atau grafik kode QR. Kemudian Anda harus menyinkronkan perangkat dan mengaitkannya dengan pengguna IAM. Lihat bagian Contoh Baru- IAMVirtual MFADevice untuk lebih jelasnya. Untuk perangkat fisik, Anda melompati langkah pembuatan dan langsung menuju sinkronisasi perangkat dan mengaitkannya dengan pengguna.

Anda dapat melampirkan tanda ke sumber daya IAM, termasuk perangkat MFA virtual, untuk mengidentifikasi, mengatur, dan mengontrol akses ke perangkat tersebut. Anda dapat menandai perangkat MFA virtual hanya ketika Anda menggunakan AWS CLI atau AWS API.

Pengguna IAM yang menggunakan SDK atau CLI dapat mengaktifkan perangkat MFA tambahan dengan memanggil atau `EnableMFADevice`menonaktifkan perangkat MFA yang ada dengan menelepon. `DeactivateMFADevice` Untuk melakukan ini dengan sukses, mereka harus terlebih dahulu memanggil `GetSessionToken`dan mengirimkan kode MFA dengan perangkat MFA yang ada. Panggilan ini mengembalikan kredensi keamanan sementara yang kemudian dapat digunakan untuk menandatangani operasi API yang memerlukan otentikasi MFA. Untuk contoh permintaan dan respons, lihat `GetSessionToken`—kredensial sementara untuk pengguna di lingkungan yang tidak tepercaya.

Untuk membuat entitas perangkat virtual di IAM untuk mewakili perangkat MFA virtual

Perintah ini memberikan ARN untuk perangkat yang digunakan untuk menggantikan nomor seri dalam banyak perintah berikut.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API: `CreateVirtualMFADevice`

Untuk mengaktifkan perangkat MFA untuk digunakan AWS

Perintah ini menyinkronkan perangkat dengan AWS dan mengaitkannya dengan pengguna. Jika perangkat ini virtual, gunakan ARN perangkat virtual sebagai nomor seri.

> **Important**
>
> Segera kirim permintaan Anda setelah membuat kode autentikasi. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirim permintaan, perangkat MFA berhasil menghubungkan pengguna namun perangkat MFA tidak akan sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu. Jika ini terjadi, Anda dapat mensinkronisasi ulang perangkat menggunakan perintah yang dijelaskan di bawah ini.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API: `EnableMFADevice`

Untuk menonaktifkan perangkat

Gunakan perintah ini untuk memisahkan perangkat dari pengguna dan menonaktifkannya. Jika perangkat ini virtual, gunakan ARN perangkat virtual sebagai nomor seri. Anda juga harus menghapus perangkat virtual secara terpisah.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API: `DeactivateMFADevice`

Untuk mencantumkan entitas perangkat MFA virtual

Gunakan perintah ini untuk mencantumkan entitas perangkat MFA virtual.

- AWS CLI: `aws iam list-virtual-mfa-devices`
- AWS API: `ListVirtualMFADevices`

Untuk menandai perangkat MFA virtual

Gunakan perintah ini untuk menandai perangkat MFA virtual.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API: `TagMFADevice`

Untuk mencantumkan tanda perangkat MFA virtual

Gunakan perintah ini untuk mencantumkan tanda yang terpasang ke perangkat MFA virtual.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API: `ListMFADeviceTags`

Untuk menghapus tanda perangkat MFA virtual

Gunakan perintah ini untuk menghapus tanda yang terpasang ke perangkat MFA virtual.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API: `UntagMFADevice`

Untuk mensinkronkan ulang perangkat MFA

Gunakan perintah ini jika perangkat menghasilkan kode yang tidak diterima oleh AWS. Jika perangkat ini virtual, gunakan ARN perangkat virtual sebagai nomor seri.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API: `ResyncMFADevice`

Untuk menghapus entitas perangkat MFA virtual di IAM

Setelah perangkat diputus kaitannya dari pengguna, Anda dapat menghapus entitas perangkat.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API: `DeleteVirtualMFADevice`

Untuk memulihkan perangkat MFA virtual yang hilang atau tidak berfungsi

Terkadang, perangkat pengguna yang meng-host aplikasi MFA virtual hilang, diganti, atau tidak berfungsi. Jika ini terjadi, pengguna tidak dapat memulihkannya sendiri. Pengguna harus menghubungi administrator untuk menonaktifkan perangkat. Lihat informasi yang lebih lengkap di Memulihkan identitas yang dilindungi MFA di IAM.

Periksa status MFA

Gunakan konsol IAM untuk memeriksa apakah pengguna IAM Pengguna root akun AWS atau IAM mengaktifkan perangkat MFA yang valid.

Untuk memeriksa status MFA dari pengguna akar

1. Masuk ke Konsol Manajemen AWS dengan kredensi pengguna root Anda dan kemudian buka konsol IAM di. https://console.aws.amazon.com/iam/
2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi keamanan.
3. Periksa di bawah Multi-factor Otentikasi (MFA) untuk melihat apakah MFA diaktifkan atau dinonaktifkan. Jika MFA belum diaktifkan, simbol peringatan ( ) akann ditampilkan

Jika Anda ingin mengaktifkan MFA untuk akun, lihat salah satu hal berikut:

- Aktifkan perangkat MFA virtual untuk pengguna root (konsol)
- Aktifkan kunci sandi atau kunci keamanan untuk pengguna root (konsol)
- Aktifkan token TOTP perangkat keras untuk pengguna root (konsol)

Untuk memeriksa status MFA pengguna IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom MFA pada tabel pengguna dengan menyelesaikan langkah-langkah berikut:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Kelola Kolom, pilih MFA.
   c. (Opsional) Kosongkan kotak centang untuk judul kolom apa pun yang tidak ingin Anda tampilkan di tabel pengguna.
   d. Pilih Tutup untuk kembali ke daftar pengguna.
4. Kolom MFA kmemberi tahu Anda tentang perangkat MFA yang diaktifkan. Jika tidak ada perangkat MFA yang aktif untuk pengguna, konsol akan menampilkan None. Jika pengguna mengaktifkan perangkat MFA, kolom MFA menunjukkan jenis perangkat yang diaktifkan dengan nilai Virtual, Kunci Keamanan, Perangkat Keras, atau SMS.

> **Note**
>
> AWS mengakhiri dukungan untuk mengaktifkan otentikasi multi-faktor SMS (MFA). Kami menyarankan agar pelanggan yang memiliki pengguna IAM yang menggunakan MFA berbasis pesan teks SMS beralih ke salah satu metode alternatif berikut: perangkat MFA virtual (berbasis perangkat lunak), kunci keamanan FIDO, atau perangkat MFA perangkat keras. Anda dapat mengidentifikasi pengguna di akun Anda dengan perangkat MFA SMS yang ditetapkan. Untuk melakukannya, kunjungi konsol IAM, pilih Pengguna dari panel navigasi, dan cari pengguna dengan SMS di kolom MFA di tabel.

5. Untuk melihat informasi tambahan tentang perangkat MFA untuk pengguna, pilih nama pengguna yang status MFA ingin Anda periksa. Kemudian pilih tab Kredensial keamanan.
6. Jika tidak ada perangkat MFA yang aktif untuk pengguna, konsol menampilkan Tidak ada perangkat MFA. Tetapkan perangkat MFA untuk meningkatkan keamanan lingkungan AWS Anda di bagian otentikasi (MFA). Multi-factor Jika pengguna mengaktifkan perangkat MFA, bagian Multi-factor autentikasi (MFA) menunjukkan detail tentang perangkat:

   - Nama perangkat
   - Jenis perangkat
   - Pengenal untuk perangkat, seperti nomor seri untuk perangkat fisik atau ARN AWS untuk perangkat virtual
   - Saat perangkat dibuat

Untuk menghapus atau menyinkronkan ulang perangkat, pilih tombol radio di sebelah perangkat dan pilih Hapus atau Sinkronkan Ulang.

Untuk informasi lebih lanjut dalam mengaktifkan MFA, lihat hal berikut:

- Tetapkan perangkat MFA virtual di Konsol Manajemen AWS
- Tetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS
- Tetapkan token TOTP perangkat keras di Konsol Manajemen AWS

Sinkronisasi ulang perangkat MFA virtual dan perangkat keras

Anda dapat menggunakan AWS untuk menyinkronkan ulang perangkat otentikasi multi-faktor (MFA) virtual dan perangkat keras Anda. Jika perangkat Anda tidak disinkronkan saat Anda mencoba

menggunakannya, upaya masuk gagal dan IAM meminta Anda untuk mensinkronkan ulang perangkat tersebut.

> **Note**
>
> Kunci keamanan FIDO tidak hilang sinkron. Jika kunci keamanan FIDO hilang atau rusak, Anda dapat menonaktifkannya. Untuk petunjuk cara menonaktifkan jenis perangkat MFA, lihat Untuk menonaktifkan perangkat MFA untuk pengguna IAM lainnya (konsol).

Sebagai AWS administrator, Anda dapat menyinkronkan ulang perangkat MFA virtual dan perangkat keras pengguna IAM jika mereka keluar dari sinkronisasi.

Jika perangkat Pengguna root akun AWS MFA Anda tidak berfungsi, Anda dapat menyinkronkan ulang perangkat menggunakan konsol IAM dengan atau tanpa menyelesaikan proses masuk. Jika Anda tidak berhasil menyinkronkan ulang perangkat Anda, Anda mungkin perlu menghapus asosiasi dan mengaitkannya kembali. Untuk informasi lebih lanjut tentang cara melakukan ini, lihat Nonaktifkan perangkat MFA danAWS Multi-factor otentikasi di IAM.

Topik

- Izin diperlukan
- Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (konsol IAM)
- Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (AWS CLI)
- Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (AWS API)

Izin diperlukan

Untuk menyinkronkan ulang perangkat MFA virtual atau perangkat keras untuk pengguna IAM Anda sendiri, Anda harus memiliki izin dari kebijakan berikut. Kebijakan ini tidak mengizinkan Anda untuk membuat atau menonaktifkan perangkat.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (konsol IAM)

Anda dapat menggunakan konsol IAM untuk mengsinkronisasi ulang perangkat MFA virtual dan perangkat keras.

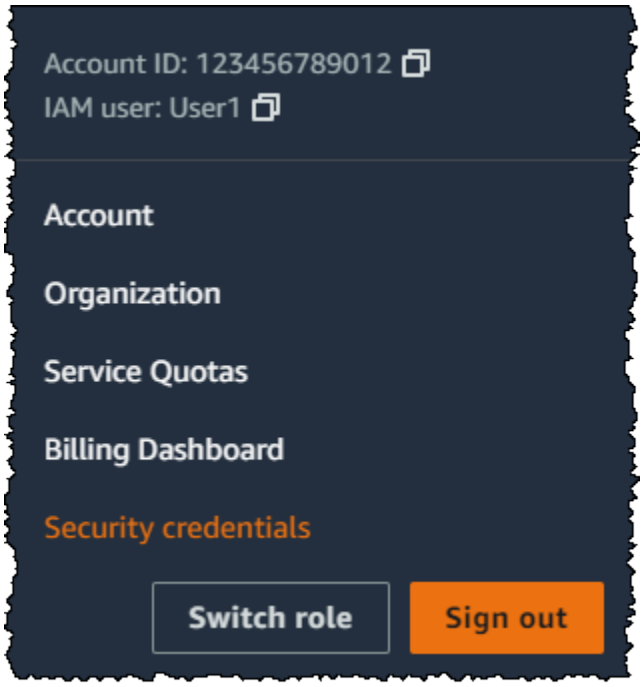
Untuk mensinkronkan perangkat keras atau MFA virtual untuk pengguna IAM Anda sendiri (konsol)

1. Gunakan ID AWS akun atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol IAM.

> **Note**
>
> Untuk kenyamanan Anda, halaman AWS masuk menggunakan cookie browser untuk mengingat nama pengguna dan informasi akun IAM Anda. Jika Anda sebelumnya masuk sebagai pengguna yang berbeda, pilih Masuk ke akun lain dekat bagian bawah halaman untuk kembali ke halaman masuk utama. Dari sana, Anda dapat mengetikkan ID AWS akun atau alias akun untuk dialihkan ke halaman masuk pengguna IAM untuk akun Anda.

Untuk mendapatkan Akun AWS ID Anda, hubungi administrator Anda.

2. Di bilah navigasi di kanan atas, pilih nama pengguna Anda, lalu pilih Kredensi bilitas keamanan.



3. Pada tab kreden AWS SIAL IAM, di bagian Multi-factor otentikasi (MFA), pilih tombol radio di sebelah perangkat MFA dan pilih Resync.
4. Ketik dua kode berikutnya yang dibuat secara berurutan dari perangkat ke dalam Kode MFA 1 dan Kode MFA 2. Kemudian pilih Resync.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan tersebut muncul

untuk bekerja tetapi perangkat tetap tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu.

Untuk mensinkronisasi ulang perangkat MFA virtual atau perangkat keras untuk pengguna IAM lainnya (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna, lalu pilih nama pengguna yang perangkat MFA perlu disinkronisasi ulang.
3. Pilih tab Kredensial keamanan. Di bagian Multi-factor otentikasi (MFA), pilih tombol radio di sebelah perangkat MFA dan pilih Resync.
4. Ketik dua kode berikutnya yang dibuat secara berurutan dari perangkat ke dalam Kode MFA 1 dan Kode MFA 2. Kemudian pilih Resync.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan tersebut muncul untuk bekerja tetapi perangkat tetap tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis-waktu kedaluwarsa setelah beberapa waktu.

Untuk mensinkronkan ulang MFA pengguna akar Anda sebelum masuk (konsol)

1. Di halaman Masuk Amazon Web Services Dengan Perangkat Autentikasi, pilih Ada masalah dengan perangkat autentikasi Anda? Klik di sini.

> **Note**
>
> Anda mungkin melihat teks yang berbeda, seperti Masuk menggunakan MFA dan Atasi masalah perangkat autentikasi Anda. Namun, fitur yang sama disediakan.

2. Di bagian Re-Sync Dengan Server Kami, ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam kode MFA 1 dan kode MFA 2. Kemudian pilih perangkat Re-sync otentikasi.

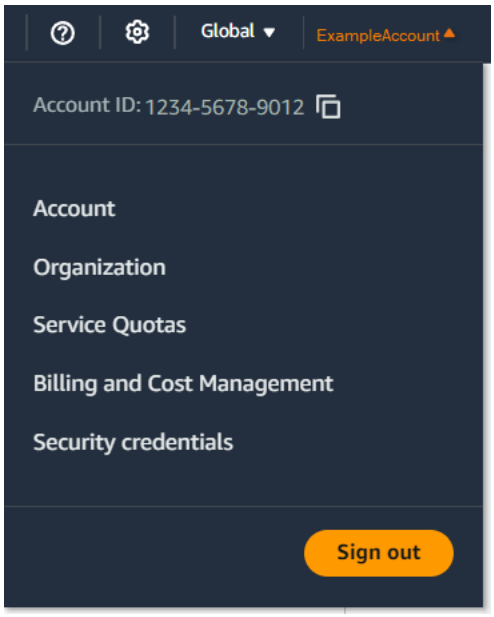
3. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk. Lalu selesaikan proses masuk menggunakan perangkat MFA Anda.

Untuk mensinkronkan ulang MFA pengguna akar Anda setelah masuk (konsol)

1. Masuk ke konsol IAM sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

> **Note**
>
> Sebagai pengguna root, Anda tidak dapat masuk ke halaman Mas uk sebagai pengguna IAM. Jika Anda melihat halaman Masuk sebagai pengguna IAM, pilih Masuk menggunakan email pengguna root di dekat bagian bawah halaman. Untuk bantuan masuk sebagai pengguna root, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.

2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi bilitas keamanan. Jika perlu, pilih L anjutkan ke kredenSIAL Keamanan.



3. Perluas bagian Multi-factor otentikasi (MFA) pada halaman.
4. Pilih tombol radio di sebelah perangkat dan pilih Resync.
5. Di kotak dialog Resync MFA device, ketik dua kode berikutnya yang dihasilkan secara berurutan dari perangkat ke dalam kode MFA 1 dan kode MFA 2. Kemudian pilih Resync.

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk membuat mengirimkan permintaan, perangkat MFA berhasil dikaitkan dengan pengguna, tetapi perangkat MFA tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat.

Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (AWS CLI)

Anda dapat menyinkronkan ulang perangkat MFA virtual dan perangkat keras dari AWS CLI.

Untuk menyinkronkan ulang perangkat MFA virtual atau perangkat keras untuk pengguna IAM (AWS CLI)

Dengan perintah perintah, terbitkan perintah aws iam resync-mfa-device:

- Perangkat MFA virtual: Menentukan Amazon Resource Name (ARN) dari perangkat sebagai nomor seri.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Perangkat MFA perangkat keras: Menentukan nomor seri perangkat keras sebagai nomor seri. Formatnya khusus vendor. Misalnya, Anda dapat membeli token gemalto dari Amazon. Nomor serinya biasa terdiri atas empat huruf, diikuti empat angka.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Kirimkan permintaan Anda segera setelah membuat kode. Jika Anda membuat kode lalu menunggunya terlalu lama untuk mengirimkan permintaan, permintaan gagal karena kode kedaluwarsa setelah beberapa saat.

Mensinkronisasi ulang perangkat MFA virtual dan perangkat keras (AWS API)

IAM memiliki panggilan API yang melakukan sinkronisasi. Dalam hal ini, kami menyarankan agar Anda memberikan izin pengguna perangkat MFA virtual dan perangkat keras untuk mengakses panggilan API ini. Kemudian buat alat berdasarkan panggilan API sehingga pengguna Anda dapat mensinkronisasi ulang perangkat mereka kapan pun diperlukan.

Untuk menyinkronkan ulang perangkat MFA virtual atau perangkat keras untuk pengguna IAM (AWS API)

- Mengirim permintaan ResyncMFADevice.

Nonaktifkan perangkat MFA

Jika Anda mengalami masalah saat masuk dengan perangkat otentikasi multi-faktor (MFA) sebagai pengguna IAM, hubungi administrator Anda untuk mendapatkan bantuan.

Sebagai administrator, Anda dapat menonaktifkan perangkat untuk pengguna IAM lainnya. Ini memungkinkan pengguna masuk tanpa menggunakan MFA. Anda dapat melakukannya sebagai solusi sementara saat perangkat MFA diganti, atau jika perangkat tidak tersedia untuk sementara. Namun, kami menyarankan Anda untuk mengaktifkan perangkat baru untuk pengguna sesegera mungkin. Untuk mempelajari cara mengaktifkan perangkat MFA baru, lihat AWS Multi-factor otentikasi di IAM.

> **Note**
>
> Jika Anda menggunakan API atau AWS CLI menghapus pengguna dari Anda Akun AWS, Anda harus menonaktifkan atau menghapus perangkat MFA pengguna. Anda membuat perubahan ini sebagai bagian dari proses penghapusan pengguna. Untuk informasi selengkapnya tentang menghapus pengguna, lihatMenghapus atau menonaktifkan pengguna IAM.

Topik

- Menonaktifkan perangkat MFA (konsol)
- Menonaktifkan perangkat MFA (AWS CLI)
- Menonaktifkan perangkat MFA (AWS API)

Menonaktifkan perangkat MFA (konsol)

Untuk menonaktifkan perangkat MFA untuk pengguna IAM lainnya (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Untuk menonaktifkan perangkat MFA bagi pengguna, pilih nama pengguna yang MFAnya ingin Anda hapus.
4. Pilih tab Kredensial keamanan.
5. Di bawah Multi-factor otentikasi (MFA), pilih tombol radio di sebelah perangkat MFA, pilih Hapus, lalu pilih Hapus.

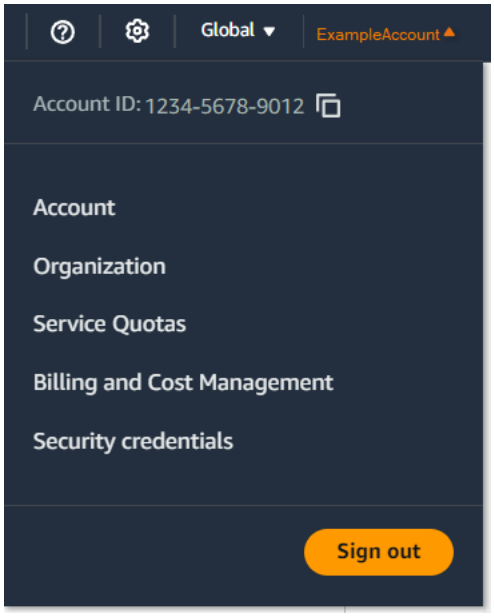
   Perangkat dihapus dari AWS. Ini tidak dapat digunakan untuk masuk atau mengotentikasi permintaan sampai diaktifkan kembali dan dikaitkan dengan AWS pengguna atau. Pengguna root akun AWS

Untuk menonaktifkan perangkat MFA untuk Anda Pengguna root akun AWS (konsol)

1. Masuk ke konsol IAM sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   > **Note**
   >
   > Sebagai pengguna root, Anda tidak dapat masuk ke halaman pengguna Masuk sebagai IAM. Jika Anda melihat halaman Masuk sebagai pengguna IAM, pilih Masuk menggunakan email pengguna root di dekat bagian bawah halaman. Untuk bantuan masuk sebagai pengguna root, lihat Masuk ke Konsol Manajemen AWS sebagai pengguna root di Panduan AWS Sign-In Pengguna.

2. Di sisi kanan bilah navigasi, pilih nama akun Anda, lalu pilih Kredensi keamanan. Jika perlu, pilih Continue to Security credentials.



3. Di bagian Multi-factor otentikasi (MFA), pilih tombol radio di sebelah perangkat MFA yang ingin Anda nonaktifkan dan pilih Hapus.
4. Pilih Hapus.

   Perangkat MFA dinonaktifkan untuk file. Akun AWS Periksa email yang terkait dengan Anda Akun AWS untuk pesan konfirmasi dari Amazon Web Services. Email tersebut memberi tahu Anda bahwa Autentikasi Multi-Faktor (MFA) Amazon Web Services Anda telah dinonaktifkan. Pesan akan datang dari `@amazon.com` atau`@aws.amazon.com`.

> **Note**
>
> Perangkat MFA virtual yang tidak ditetapkan di Akun AWS Anda akan dihapus saat Anda menambahkan perangkat MFA virtual baru baik melalui atau selama Konsol Manajemen AWS proses masuk. Perangkat MFA virtual yang tidak ditetapkan adalah perangkat di akun Anda tetapi tidak digunakan oleh pengguna root akun atau pengguna IAM untuk proses masuk. Mereka dihapus sehingga perangkat MFA virtual baru dapat ditambahkan ke akun Anda. Ini juga memungkinkan Anda untuk menggunakan kembali nama perangkat.

Menonaktifkan perangkat MFA (AWS CLI)

Untuk menonaktifkan perangkat MFA untuk pengguna IAM (AWS CLI)

- Jalankan perintah ini: `aws iam deactivate-mfa-device`

Menonaktifkan perangkat MFA (AWS API)

Untuk menonaktifkan perangkat MFA untuk pengguna IAM (AWS API)

- Hubungi operasi ini: `DeactivateMFADevice`

Memulihkan identitas yang dilindungi MFA di IAM

Jika perangkat MFA virtual atau token TOTP perangkat keras Anda tampaknya berfungsi dengan baik, tetapi Anda tidak dapat menggunakannya untuk mengakses AWS sumber daya, itu mungkin tidak disinkronkan dengan. AWS Untuk informasi tentang sinkronisasi perangkat MFA virtual atau MFA perangkat keras, lihat Sinkronisasi ulang perangkat MFA virtual dan perangkat keras. Kunci keamanan FIDO tidak hilang sinkron.

Jika perangkat ??? MFA Pengguna root akun AWS hilang, rusak, atau tidak berfungsi, Anda dapat memulihkan akses ke akun Anda. Pengguna IAM harus menghubungi administrator untuk menonaktifkan perangkat.

> **Important**
>
> Sebaiknya aktifkan beberapa perangkat MFA. Mendaftarkan beberapa perangkat MFA membantu memastikan akses berkelanjutan jika perangkat hilang atau rusak. Pengguna Anda Pengguna root akun AWS dan IAM dapat mendaftarkan hingga delapan perangkat MFA jenis apa pun.

Prasyarat — Gunakan perangkat MFA lain

Jika perangkat otentikasi multi-faktor (MFA) hilang, rusak, atau tidak berfungsi, Anda dapat masuk menggunakan perangkat MFA lain yang terdaftar pada pengguna root atau pengguna IAM yang sama.

Cara masuk menggunakan perangkat MFA lain

1. Masuk ke Konsol Manajemen AWS dengan Akun AWS ID atau alias akun dan kata sandi Anda.
2. Pada halaman Veri fikasi tambahan yang diperlukan atau halaman Multi-factor otentikasi, pilih C oba metode MFA lain.
3. Otentikasi dengan jenis perangkat MFA yang Anda pilih.

4. Langkah selanjutnya bervariasi berdasarkan apakah Anda berhasil masuk dengan perangkat MFA alternatif.
   - Jika Anda berhasil masuk, Anda bisaSinkronisasi ulang perangkat MFA virtual dan perangkat keras, yang dapat menyelesaikan masalah. Jika perangkat MFA Anda hilang atau rusak, Anda dapat menonaktifkannya. Untuk petunjuk cara menonaktifkan jenis perangkat MFA, lihat Nonaktifkan perangkat MFA.
   - Jika Anda tidak dapat masuk dengan MFA, gunakan langkah-langkah dalam Pemulihan perangkat MFA pengguna akar atau Pemulihan perangkat MFA pengguna IAM untuk memulihkan identitas yang dilindungi MFA Anda.

Pemulihan perangkat MFA pengguna akar

Jika Anda tidak dapat masuk dengan MFA, Anda dapat menggunakan metode otentikasi alternatif untuk masuk dengan memverifikasi identitas Anda menggunakan email dan nomor telepon kontak utama yang terdaftar di akun Anda.

Konfirmasikan Anda dapat mengakses email dan nomor telepon kontak utama yang terkait dengan akun Anda sebelum Anda menggunakan faktor otentikasi alternatif untuk masuk sebagai pengguna root. Jika Anda perlu memperbarui nomor telepon kontak utama, masuk sebagai pengguna IAM dengan akses Administrator alih-alih pengguna root. Untuk petunjuk tambahan tentang memperbarui informasi kontak akun, lihat Mengedit informasi kontak di Panduan AWS Billing Pengguna. Jika Anda tidak memiliki akses ke email dan nomor telepon kontak utama, Anda harus menghubungi AWS Dukungan.

> **Important**
>
> Sebaiknya perbarui alamat email dan nomor telepon kontak yang ditautkan ke pengguna root agar pemulihan akun berhasil. Untuk informasi selengkapnya, lihat Memper barui kontak utama untuk Anda Akun AWS di Panduan AWS Account Management Referensi.

Untuk masuk menggunakan faktor otentikasi alternatif sebagai Pengguna root akun AWS

1. Masuk ke Konsol Manajemen AWS sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.
2. Pada halaman Veri fikasi tambahan yang diperlukan, pilih metode MFA untuk diautentikasi dan pilih Berikutnya.

> **Note**
>
> Anda mungkin melihat teks alternatif, seperti Masuk menggunakan MFA, Memecahkan masalah perangkat otentikasi Anda, atau Mem ecahkan masalah MFA, tetapi fungsinya sama. Jika Anda tidak dapat menggunakan faktor otentikasi alternatif untuk memverifikasi alamat email akun dan nomor telepon kontak utama, hubungi AWS Dukungan untuk menonaktifkan perangkat MFA Anda.

3. Tergantung pada jenis MFA yang Anda gunakan, Anda akan melihat halaman yang berbeda, tetapi opsi Troubles hooting MFA berfungsi sama. Pada halaman verifikasi tambahan yang diperlukan atau halaman Multi-factor otentikasi, pilih Memecahkan masalah MFA.
4. Jika perlu, ketikkan kata sandi Anda kembali dan pilih Masuk.
5. Pada halaman Memecahkan masalah perangkat otentikasi Anda, di bagian Masuk menggunakan faktor otentikasi alternatif, pilih Masuk menggunakan faktor alternatif.
6. Pada halaman Masuk menggunakan faktor otentikasi alternatif, autentikasi akun Anda dengan memverifikasi alamat email, pilih Kirim email verifikasi.
7. Periksa email yang terkait dengan Anda Akun AWS untuk pesan dari Amazon Web Services (recover-mfa-no-reply@verify.signin.aws). Ikuti petunjuk di dalam email.

   Jika Anda tidak melihat email di akun Anda, periksa folder spam, atau kembali ke peramban Anda dan pilih Kirim ulang email.
8. Setelah Anda memverifikasi alamat email, Anda dapat melanjutkan mengautentikasi akun Anda. Untuk memverifikasi nomor telepon kontak utama Anda, pilih Hu bungi saya sekarang.
9. Jawab panggilan dari AWS dan, ketika diminta, masukkan nomor 6 digit dari AWS situs web di keypad ponsel Anda.

   Jika Anda tidak menerima panggilan dari AWS, pilih Mas uk untuk masuk ke konsol lagi dan memulai dari awal. Atau lihat Perangkat Oten Multi-Factor tikasi (MFA) yang hilang atau tidak dapat digunakan untuk menghubungi dukungan untuk meminta bantuan.
10. Setelah memverifikasi nomor telepon, Anda dapat masuk ke akun dengan memilih Masuk ke konsol.
11. Langkah berikutnya bervariasi tergantung pada jenis MFA yang Anda gunakan:

- Untuk perangkat MFA virtual, hapus akun dari perangkat Anda. Lalu, kunjungi Kredensial Keamanan AWS dan hapus perangkat virtual MFA lama sebelum Anda membuat perangkat baru.
- Untuk kunci keamanan FIDO, lihat halaman KredenSIAL AWS Keamanan dan nonaktifkan kunci keamanan FIDO lama sebelum mengaktifkan yang baru.
- Untuk token TOTP perangkat keras, hubungi penyedia pihak ketiga untuk bantuan memperbaiki atau mengganti perangkat. Anda dapat terus masuk menggunakan faktor alternatif autentikasi hingga Anda menerima perangkat baru. Setelah Anda memiliki perangkat MFA perangkat keras baru, lihat halaman KredenSIAL AWS Keamanan dan hapus perangkat MFA lama.

> **Note**
>
> Anda tidak perlu mengganti perangkat MFA yang hilang atau dicuri dengan jenis perangkat yang sama. Misalnya, jika Anda memecahkan kunci keamanan FIDO Anda dan memesan yang baru, Anda dapat menggunakan MFA virtual atau token TOTP perangkat keras hingga kunci FIDO baru tiba.

> **Important**
>
> Jika perangkat MFA Anda hilang atau dicuri, ubah kata sandi pengguna root Anda setelah masuk dan membuat perangkat MFA pengganti Anda. Penyerang mungkin telah mencuri perangkat otentikasi dan mungkin juga memiliki kata sandi Anda saat ini. Untuk informasi selengkapnya, lihat Ubah kata sandi untuk Pengguna root akun AWS.

Pemulihan perangkat MFA pengguna IAM

Jika Anda adalah pengguna IAM yang tidak dapat masuk dengan MFA, Anda tidak dapat memulihkan perangkat MFA sendiri. Anda harus menghubungi administrator untuk menonaktifkan perangkat. Kemudian Anda dapat mengaktifkan perangkat baru.

Untuk mendapatkan bantuan bagi perangkat MFA sebagai pengguna IAM

1. Hubun AWS gi administrator atau orang lain yang memberi Anda nama pengguna dan kata sandi untuk pengguna IAM. Administrator harus menonaktifkan perangkat MFA seperti yang dijelaskan dalam Nonaktifkan perangkat MFA sehingga Anda dapat masuk.
2. Langkah berikutnya bervariasi tergantung pada jenis MFA yang Anda gunakan:
   - Untuk perangkat MFA virtual, hapus akun dari perangkat Anda. Kemudian aktifkan perangkat virtual seperti yang dijelaskan di Tetapkan perangkat MFA virtual di Konsol Manajemen AWS.
   - Untuk kunci keamanan FIDO, hubungi penyedia pihak ketiga untuk bantuan mengganti perangkat. Saat Anda menerima kunci keamanan FIDO baru, aktifkan seperti yang dijelaskan dalamTetapkan kunci sandi atau kunci keamanan di Konsol Manajemen AWS.
   - Untuk token TOTP perangkat keras, hubungi penyedia pihak ketiga untuk bantuan memperbaiki atau mengganti perangkat. Setelah Anda memiliki perangkat MFA fisik baru, aktifkan perangkat seperti yang dijelaskan di Tetapkan token TOTP perangkat keras di Konsol Manajemen AWS.

   > **Note**
   >
   > Anda tidak perlu mengganti perangkat MFA yang hilang atau dicuri dengan jenis perangkat yang sama. Anda dapat memiliki hingga delapan perangkat MFA dari kombinasi apa pun. Misalnya, jika Anda memecahkan kunci keamanan FIDO Anda dan memesan yang baru, Anda dapat menggunakan MFA virtual atau token TOTP perangkat keras hingga kunci FIDO baru tiba.
3. Jika perangkat MFA Anda hilang atau dicuri, juga ubah kata sandi Anda apabila penyerang telah mencuri perangkat autentikasi dan mungkin juga memiliki kata sandi Anda saat ini. Untuk informasi selengkapnya, lihat Kelola kata sandi untuk pengguna IAM

Akses API aman dengan MFA

Dengan kebijakan IAM, Anda dapat menentukan operasi API mana yang diizinkan untuk dipanggil oleh pengguna. Anda dapat menerapkan keamanan tambahan dengan mewajibkan pengguna untuk melakukan otentikasi dengan otentikasi multi-faktor (MFA) sebelum mengizinkan mereka melakukan tindakan yang sangat sensitif.

Misalnya, Anda mungkin memiliki kebijakan yang memungkinkan pengguna untuk melakukan Amazon EC2 `RunInstances`, `DescribeInstances`, dan `StopInstances`. Tetapi Anda mungkin ingin membatasi tindakan destruktif seperti `TerminateInstances` dan memastikan bahwa pengguna dapat melakukan tindakan itu hanya jika mereka melakukan otentikasi dengan perangkat AWS MFA.

Topik

- Gambaran umum
- Skenario: Perlindungan MFA untuk pendelegasian lintas-akun
- Skenario: Perlindungan MFA untuk akses ke operasi API di akun saat ini
- Skenario: Perlindungan MFA untuk sumber daya yang memiliki kebijakan berbasis sumber daya

Gambaran umum

Menambahkan perlindungan MFA ke operasi API melibatkan tugas-tugas ini:

1. Administrator mengonfigurasi perangkat AWS MFA untuk setiap pengguna yang harus membuat permintaan API yang memerlukan otentikasi MFA. Untuk informasi selengkapnya, lihat AWS Multi-factor otentikasi di IAM.
2. Administrator membuat kebijakan untuk pengguna yang menyertakan `Condition` elemen yang memeriksa apakah pengguna diautentikasi dengan perangkat AWS MFA.
3. Pengguna memanggil salah satu operasi AWS STS API yang mendukung parameter MFA: AssumeRole atau GetSessionToken. Sebagai bagian dari panggilan, pengguna mencakup pengidentifikasi perangkat untuk perangkat yang terhubung dengan pengguna. Pengguna juga menyertakan kata sandi satu kali berbasis waktu (TOTP) yang dihasilkan perangkat. Dalam kasus lain, pengguna mendapatkan kembali kredensial keamanan sementara yang dapat digunakan pengguna untuk membuat permintaan tambahan ke AWS.

> **Note**
>
> Perlindungan MFA untuk operasi API layanan hanya tersedia jika layanan mendukung kredensial keamanan sementara. Untuk daftar layanan ini, lihat Menggunakan Kredensial IT Sementara untuk Mengakses AWS.

Jika otorisasi gagal, AWS mengembalikan pesan kesalahan akses ditolak (seperti halnya untuk akses yang tidak sah). Dengan kebijakan MFA-protected API, menolak akses AWS ke operasi API yang ditentukan dalam kebijakan jika pengguna mencoba memanggil operasi API tanpa otentikasi MFA yang valid. Operasi juga ditolak jika stempel waktu permintaan untuk operasi API berada di luar rentang yang diizinkan yang ditentukan dalam kebijakan. Pengguna harus diautentikasi ulang menggunakan MFA dengan meminta kredensial keamanan sementara yang baru, dengan kode MFA dan nomor seri perangkat.

Kebijakan IAM dengan ketentuan MFA

Kebijakan dengan ketentuan MFA dapat dilampirkan ke hal berikut:

- Pengguna atau grup IAM
- Sumber daya seperti bucket Amazon S3, antrean Amazon Amazon SQS, atau topik Amazon SNS
- Kebijakan kepercayaan dari peran IAM yang dapat diasumsikan oleh pengguna

Anda dapat menggunakan ketentuan MFA dalam kebijakan untuk memeriksa properti berikut:

- Ekstensi—Untuk memudahkan verifikasi bahwa pengguna melakukan verifikasi dengan MFA, periksa apakah kunci `aws:MultiFactorAuthPresent` adalah `True` dalam ketentuan `Bool`. Kuncinya hanya ada ketika pengguna mengotentikasi dengan kredenSIAL jangka pendek. Long-term kredenSIAL, seperti kunci akses, tidak menyertakan kunci ini.
- Durasi—Jika Anda ingin memberikan akses hanya untuk waktu yang ditentukan setelah otentikasi MFA, gunakan jenis numerik untuk membandingkan usia kunci `aws:MultiFactorAuthAge` i ke nilai (seperti 3600 detik). Perhatikan bahwa kunci `aws:MultiFactorAuthAge` tidak ada jika MFA tidak digunakan.

Contoh berikut menunjukkan kebijakan kepercayaan peran IAM yang menyertakan ketentuan MFA untuk menguji keberadaan autentikasi MFA. Dengan kebijakan ini, pengguna dari `Principal` elemen yang Akun AWS ditentukan (ganti `ACCOUNT-B-ID` dengan Akun AWS ID yang valid) dapat mengambil peran yang dilampirkan kebijakan ini. Namun, pengguna tersebut hanya dapat mengasumsikan peran jika pengguna diautentikasi menggunakan MFA.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Untuk informasi lebih lanjut tentang jenis ketentuan untuk MFA, lihat AWS kunci konteks kondisi global, Operator ketentuan numerik, dan Operator ketentuan memeriksa keberadaan kunci kondisi.

Memilih antara GetSessionToken dan AssumeRole

AWS STS menyediakan dua operasi API yang memungkinkan pengguna meneruskan informasi MFA: `GetSessionToken` dan`AssumeRole`. Operasi API yang dipanggil pengguna untuk mendapatkan kredensial keamanan sementara bergantung pada skenario mana yang berlaku.

Gunakan **`GetSessionToken`** untuk skenario berikut:

- Panggil operasi API yang mengakses sumber daya Akun AWS sama dengan pengguna IAM yang membuat permintaan. Perhatikan bahwa kredenSIAL sementara dari `GetSessionToken` permintaan dapat mengakses operasi IAM dan AWS STS API hanya jika Anda menyertakan informasi MFA dalam permintaan kredenSIAL. Karena kredensial sementara dikembalikan oleh `GetSessionToken` meliputi informasi MFA, Anda dapat memeriksa MFA dalam operasi API individu yang dilakukan oleh kredensial.
- Akses ke sumber daya yang dilindungi dengan kebijakan berbasis sumber daya yang mencakup ketentuan MFA.

Tujuan dari operasi `GetSessionToken` adalah mengautentikasi pengguna menggunakan MFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol operasi autentikasi.

Gunakan **`AssumeRole`** untuk skenario berikut:

- Panggil operasi API yang mengakses sumber daya yang sama atau berbeda Akun AWS. Panggilan API dapat menyertakan IAM atau AWS STS API apa pun. Perhatikan bahwa untuk melindungi akses Anda, Anda memberlakukan MFA pada saat pengguna menjalankan peran tersebut. Kredensial sementara dikembalikan oleh `AssumeRole` tidak menyertakan informasi MFA dalam konteks, jadi Anda tidak dapat memeriksa operasi API individu untuk MFA. Inilah mengapa

Anda harus menggunakan `GetSessionToken` untuk membatasi akses ke sumber daya yang dilindungi oleh kebijakan berbasis sumber daya.

> **Note**
>
> AWS CloudTrail log akan berisi informasi MFA ketika pengguna IAM masuk dengan MFA. Jika pengguna IAM mengambil peran IAM, juga CloudTrail akan `mfaAuthenticated: true` masuk `sessionContext` atribut untuk tindakan yang dilakukan menggunakan peran yang diasumsikan. Namun, CloudTrail logging terpisah dari apa yang diperlukan IAM ketika panggilan API dilakukan dengan kredentif peran yang diasumsikan. Untuk informasi selengkapnya, lihat CloudTrail UserIdentity Element.

Perincian tentang cara menerapkan skenario ini akan dijelaskan nanti dalam dokumen ini.

Poin penting tentang akses MFA-protected API

Penting untuk memahami aspek perlindungan MFA berikut untuk operasi API:

- Perlindungan MFA hanya tersedia dalam bentuk kredensial keamanan sementara, yang harus diperoleh dengan `AssumeRole` atau `GetSessionToken`.
- Anda tidak dapat menggunakan akses MFA-protected API dengan Pengguna root akun AWS kredenSIAL.
- Anda tidak dapat menggunakan akses MFA-protected API dengan kunci keamanan U2F.
- Pengguna federasi tidak dapat ditetapkan perangkat MFA untuk digunakan dengan AWS layanan, sehingga mereka tidak dapat mengakses AWS sumber daya yang dikendalikan oleh MFA. (Lihat poin berikutnya.)
- Oper AWS STS asi API lain yang mengembalikan kredenSIAL sementara tidak mendukung MFA. Untuk `AssumeRoleWithWebIdentity` dan`AssumeRoleWithSAML`, pengguna diautentikasi oleh penyedia eksternal dan AWS tidak dapat menentukan apakah penyedia tersebut memerlukan MFA. Untuk `GetFederationToken`, MFA tidak selalu berkaitan dengan pengguna tertentu.
- Demikian pula, kredenSIAL jangka panjang (kunci akses pengguna IAM dan kunci akses pengguna root) tidak dapat digunakan dengan akses MFA-protected API karena tidak kedaluwarsa.
- `AssumeRole` dan `GetSessionToken` juga dapat dipanggil tanpa informasi MFA. Dalam hal ini, penelepon mendapatkan kembali kredensial keamanan sementara, tetapi informasi sesi untuk kredensial sementara tersebut tidak menunjukkan bahwa pengguna mengautentikasi dengan MFA.

- Untuk menetapkan perlindungan MFA untuk operasi API, Anda menambahkan ketentuan MFA ke kebijakan. Kebijakan harus mencantumkan kunci keamanan `aws:MultiFactorAuthPresent` untuk menerapkan penggunaan MFA. Untuk pendelegasian lintas akun, kebijakan kepercayaan peran tersebut harus mencakup kunci ketentuan.
- Ketika Anda mengiz Akun AWS inkan orang lain untuk mengakses sumber daya di akun Anda, keamanan sumber daya Anda tergantung pada konfigurasi akun tepercaya (akun lain, bukan milik Anda). Hal ini tetap berlaku meskipun Anda memerlukan autentikasi multi-faktor. Setiap identitas dalam akun tepercaya yang memiliki izin untuk membuat perangkat MFA virtual dapat membuat klaim MFA untuk memenuhi bagian dari kebijakan kepercayaan peran Anda. Sebelum mengizinkan anggota akun lain mengakses AWS sumber daya yang memerlukan otentikasi multi-faktor, Anda harus memastikan bahwa pemilik akun tepercaya mengikuti praktik terbaik keamanan. Misalnya, akun tepercaya harus membatasi akses ke operasi API sensitif, seperti operasi API manajemen perangkat MFA, secara spesifik, identitas terpercaya.
- Jika kebijakan mencakup ketentuan MFA, permintaan ditolak jika pengguna belum menjadi MFA yang diautentikasi, atau jika mereka memberikan pengidentifikasi perangkat MFA yang tidak valid atau TOTP yang tidak valid.

Skenario: Perlindungan MFA untuk pendelegasian lintas-akun

Dalam skenario ini, Anda ingin mendelegasikan akses ke pengguna IAM di akun lain, tetapi hanya jika pengguna diautentikasi dengan perangkat AWS MFA. Untuk informasi selengkapnya tentang delegasi lintas akun, lihatIstilah dan konsep peran.

Bayangkan Anda memiliki akun A (akun kepercayaan yang memiliki sumber daya yang akan diakses), dengan pengguna IAM Anaya, yang memiliki izin administrator. Dia ingin memberikan akses ke pengguna Richard di akun B (akun tepercaya), tetapi ingin memastikan bahwa Richard terotentikasi dengan MFA sebelum dia mengambil peran tersebut.

1. Di akun kepercayaan A, Anaya membuat peran IAM bernama `CrossAccountRole` dan menetapkan prinsip dalam kebijakan kepercayaan peran ke ID akun B. Kebijakan kepercayaan memberikan izin untuk tindakan tersebut AWS STS `AssumeRole`. Anaya juga menambahkan ketentuan MFA ke kebijakan kepercayaan, seperti dalam contoh berikut.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
```

```
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya menambahkan kebijakan izin untuk peran yang menentukan apa yang dapat dilakukan peran tersebut. Kebijakan izin untuk peran dengan perlindungan MFA tidak berbeda dengan kebijakan izin peran lainnya. Contoh berikut menunjukkan kebijakan yang ditambahkan Anaya ke peran; kebijakan ini memungkinkan pengguna asumsi untuk melakukan tindakan Amazon DynamoDB apa pun pada tabel `Books` di akun A. Kebijakan ini juga mengizinkan `dynamodb:ListTables` tindakan, yang diperlukan untuk melakukan tindakan di konsol.

> **Note**
>
> Kebijakan izin tidak mencakup ketentuan MFA. Penting untuk memahami bahwa autentikasi MFA hanya digunakan untuk menentukan apakah pengguna dapat menjalankan peran tersebut. Setelah pengguna mengambil peran tersebut, tidak ada lagi pemeriksaan MFA yang dilakukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:111122223333:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
```

```
}
```

3. Di akun tepercaya B, administrator memastikan bahwa pengguna IAM Richard dikonfigurasi dengan perangkat AWS MFA dan bahwa ia mengetahui ID perangkat. ID perangkat adalah nomor seri jika MFA perangkat keras, atau ARN perangkat jika itu adalah perangkat MFA virtual.
4. Di akun B, administrator melampirkan kebijakan berikut kepada pengguna Richard (atau grup yang ia anggota) yang memungkinkannya untuk menghubungi tindakan `AssumeRole`. Sumber daya diatur ke ARN dari peran yang dibuat Anaya pada langkah 1. Perhatikan bahwa kebijakan ini tidak mengandung ketentuan MFA.

   JSON

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

5. Di akun B, Richard (atau aplikasi yang dijalankan Richard) akan memanggil `AssumeRole`. Panggilan API mencakup ARN peran yang akan diasumsikan (`arn:aws:iam::`*`ACCOUNT-A-ID`*`:role/CrossAccountRole`), ID perangkat MFA, dan TOTP saat ini yang Richard dapatkan dari perangkatnya.

   Ketika Richard meneleponAssumeRole, AWS menentukan apakah dia memiliki kredensi yang valid, termasuk persyaratan untuk MFA. Jika demikian, Richard berhasil mengambil peran dan dapat melakukan tindakan DynamoDB apa pun pada tabel yang diberi nama Books di akun A saat menggunakan kredenSIAL sementara peran tersebut.

   Sebagai contoh program yang memanggil `AssumeRole`, lihat Memanggil AssumeRole dengan otentikasi MFA.

Skenario: Perlindungan MFA untuk akses ke operasi API di akun saat ini

Dalam skenario ini, Anda harus memastikan bahwa pengguna di Anda Akun AWS dapat mengakses operasi API sensitif hanya ketika pengguna diautentikasi menggunakan perangkat AWS MFA.

Bayangkan bahwa Anda memiliki akun A yang berisi sekelompok developer yang perlu bekerja dengan kasus EC2. Pengembang biasa dapat bekerja dengan instans, tetapi mereka tidak diberikan izin tindakan `ec2:StopInstances` atau `ec2:TerminateInstances`. Anda ingin membatasi tindakan istimewa yang “menghancurkan” tersebut ke hanya beberapa pengguna tepercaya, sehingga Anda menambahkan perlindungan MFA ke kebijakan yang memungkinkan tindakan Amazon EC2 yang sensitif ini.

Dalam skenario ini, salah satu pengguna tepercaya tersebut adalah pengguna Sofía. Pengguna Anaya adalah administrator di akun A.

1. Anaya memastikan bahwa Sofía dikonfigurasi dengan perangkat AWS MFA dan Sofía mengetahui ID perangkat. ID perangkat adalah nomor seri jika MFA perangkat keras, atau ARN perangkat jika itu adalah perangkat MFA virtual.
2. Anaya membuat kelompok bernama `EC2-Admins` dan menambahkan Sofía ke grup.
3. Anaya melampirkan kebijakan berikut ke kelompok `EC2-Admins`. Kebijakan ini memberikan izin kepada pengguna untuk menghubungi tindakan Amazon EC2 `StopInstances` dan `TerminateInstances` hanya jika pengguna telah mengautentikasi menggunakan MFA.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.
> **Note**
>
> Agar kebijakan ini berlaku, pengguna harus keluar terlebih dahulu dan masuk kembali.

   Jika pengguna Sofía perlu menghentikan atau menghentikan instans Amazon EC2, dia (atau aplikasi yang sedang dia jalankan) memanggilnya. `GetSessionToken` Operasi API ini meneruskan ID perangkat MFA dan TOTP saat ini yang diperoleh Sofía dari perangkatnya.
5. Pengguna Sofía (atau aplikasi yang digunakan Sofía) menggunakan kredensial sementara yang diberikan oleh `GetSessionToken` untuk menghubungi tindakan Amazon EC2 `StopInstances` atau `TerminateInstances`.

   Sebagai contoh program yang memanggil `GetSessionToken`, lihat Memanggil GetSessionToken dengan otentikasi MFA nanti di dokumen ini.

Skenario: Perlindungan MFA untuk sumber daya yang memiliki kebijakan berbasis sumber daya

Dalam skenario ini, Anda adalah pemilik bucket S3, antrean SQS, atau topik SNS. Anda ingin memastikan bahwa setiap pengguna dari Akun AWS siapa pun yang mengakses sumber daya diautentikasi oleh perangkat AWS MFA.

Skenario ini mengilustrasikan cara untuk menyediakan jangkauan perlindungan MFA lintas akun yang mengharuskan pengguna untuk menjalankan peran terlebih dahulu. Dalam hal ini, pengguna dapat mengakses sumber daya jika tiga kondisi terpenuhi: Pengguna harus diautentikasi oleh MFA, dapat memperoleh kredensial keamanan sementara dari `GetSessionToken`, dan berada di akun yang dipercaya oleh kebijakan sumber daya.

Bayangkan Anda berada di akun A dan Anda membuat bucket S3. Anda ingin memberikan akses ke bucket ini kepada pengguna yang berada di beberapa tempat berbeda Akun AWS, tetapi hanya jika pengguna tersebut diautentikasi dengan MFA.

Dalam skenario ini, pengguna Anaya adalah administrator di akun A. Pengguna Nikhil adalah pengguna IAM di akun C.

1. Pada akun A, Anaya membuat bucket dengan nama `Account-A-bucket`.
2. Anaya menambahkan kebijakan bucket ke bucket. Kebijakan ini memungkinkan pengguna di akun A, akun B, atau akun C untuk melakukan tindakan Amazon S3 `PutObject` dan `DeleteObject` di dalam bucket. Kebijakan ini mencakup ketentuan MFA.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 menawarkan fitur Hapus MFA untuk (hanya) akses akun akar. Anda dapat mengaktifkan Hapus MFA Amazon S3 saat Anda mengatur status versi bucket. Amazon S3 MFA Delete tidak dapat diterapkan ke pengguna IAM, dan dikelola secara independen dari akses MFA-protected API. Seorang pengguna IAM dengan izin untuk menghapus bucket tidak dapat menghapus bucket dengan Hapus MFA Amazon S3 yang diaktifkan. Untuk informasi selengkapnya tentang Amazon S3 MFA Delete, lihat MFA Hapus.

3. Di akun C, administrator memastikan bahwa pengguna Nikhil dikonfigurasi dengan perangkat MFA AWS dan bahwa dia mengetahui ID perangkat. ID perangkat adalah nomor seri jika MFA perangkat keras, atau ARN perangkat jika itu adalah perangkat MFA virtual.
4. Dalam akun C, Nikhil (atau aplikasi yang dia jalankan) akan memanggil `GetSessionToken`. Panggilan mencakup ID atau ARN perangkat MFA dan TOTP saat ini yang diperoleh Nikhil dari perangkatnya.
5. Nikhil (atau aplikasi yang dia gunakan) menggunakan kredensial sementara yang dikembalikan oleh `GetSessionToken` untuk menghubungi tindakan Amazon S3 `PutObject` untuk mengunggah file ke `Account-A-bucket`.

Sebagai contoh program yang memanggil `GetSessionToken`, lihat Memanggil GetSessionToken dengan otentikasi MFA nanti di dokumen ini.

> **Note**
>
> Kredensial sementara yang dikembalikan `AssumeRole` tidak akan bekerja dalam kasus ini. Meskipun pengguna dapat memberikan informasi MFA untuk mengambil peran, kredensial sementara dikembalikan oleh `AssumeRole` tidak menyertakan informasi MFA. Informasi tersebut diperlukan untuk memenuhi ketentuan MFA dalam kebijakan.

Kode sampel: Meminta kredensial dengan autentikasi multi-faktor

Contoh berikut menunjukkan cara melakukan panggilan `GetSessionToken` dan operasi `AssumeRole` dan lolos parameter autentikasi MFA. Tidak ada izin yang diperlukan untuk memanggil `GetSessionToken`, tetapi Anda harus memiliki kebijakan yang mengijinkan Anda untuk memanggil `AssumeRole`. Kredensial yang dihasilkan kemudian digunakan untuk mencantumkan semua bucket S3 di akun.

Memanggil GetSessionToken dengan otentikasi MFA

Contoh berikut menunjukkan cara memanggil `GetSessionToken` dan meneruskan informasi otentikasi MFA. Kredensial keamanan sementara yang dihasilkan oleh operasi `GetSessionToken` kemudian digunakan untuk mencantumkan semua bucket S3 di akun.

Kebijakan yang terlampir pada pengguna yang menjalankan kode ini (atau ke grup yang ada di pengguna) memberikan izin untuk kredensial sementara yang dikembalikan. Untuk contoh kode ini, kebijakan harus memberikan izin kepada pengguna untuk meminta operasi Amazon S3 `ListBuckets`.

Contoh kode berikut menunjukkan cara menggunakan`GetSessionToken`.

CLI

AWS CLI

Untuk mendapatkan satu set kredensi jangka pendek untuk identitas IAM

`get-session-token`Perintah berikut mengambil satu set kredensi jangka pendek untuk identitas IAM yang membuat panggilan. Kredensi yang dihasilkan dapat digunakan untuk

permintaan di mana otentikasi multi-faktor (MFA) diperlukan oleh kebijakan. Kredensialnya kedaluwarsa 15 menit setelah dibuat.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Untuk informasi lebih lanjut, lihat Meminta Kredensial Keamanan Sementara dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetSessionTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensial sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 3: Mengembalikan **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensi sementara yang valid selama satu jam menggunakan nomor identifikasi perangkat MFA yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang diberikan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Untuk detail API, lihat GetSessionTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensial sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 3: Mengembalikan **Amazon.RuntimeAWSCredentials** instance yang berisi kredensi sementara yang valid selama satu jam menggunakan nomor identifikasi perangkat MFA yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang diberikan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Untuk detail API, lihat GetSessionTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan token sesi dengan meneruskan token MFA dan gunakan untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
```

```
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk detail API, lihat GetSessionTokendi AWS SDK for Python (Boto3) Referensi API.

Memanggil AssumeRole dengan otentikasi MFA

Contoh berikut menunjukkan cara memanggil `AssumeRole` dan meneruskan informasi otentikasi MFA. Kredensial keamanan sementara yang dikembalikan oleh `AssumeRole` kemudian digunakan untuk mencantumkan semua bucket Amazon S3 di akun.

Untuk informasi selengkapnya tentang skenario ini, lihat Skenario: Perlindungan MFA untuk pendelegasian lintas-akun.

Contoh kode berikut menunjukkan cara menggunakan`AssumeRole`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
```

```
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
```

```
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengambil peran

`assume-role`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran IAM. `s3-access-example`

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Output:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
```

```
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Output dari perintah berisi kunci akses, kunci rahasia, dan token sesi yang dapat Anda gunakan untuk AWS mengautentikasi.

Untuk penggunaan AWS CLI, Anda dapat mengatur profil bernama yang terkait dengan peran. Saat Anda menggunakan profil, AWS CLI akan memanggil peran assume-dan mengelola kredensil untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan peran IAM di AWS CLI di Panduan Pengguna AWS CLI.

- Untuk detail API, lihat AssumeRoledi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
```

```
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
```

```
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
exTime);

        } catch (StsException e) {
```

```
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat klien.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asumsikan peran IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
```

```
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
```

```
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensi yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari MFA yang terkait dengan kredensi pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk detail API, lihat AssumeRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensi yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari MFA yang terkait dengan kredensi pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk detail API, lihat AssumeRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Asumsikan peran IAM yang memerlukan token MFA dan gunakan kredensil sementara untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
```

```
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk detail API, lihat AssumeRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Untuk detail API, lihat AssumeRolereferensi AWS SDK for Rust API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi AssumeRole AWSSDK untuk Swift API.

Kredensi khusus layanan untuk pengguna IAM

Kredensi khusus layanan adalah mekanisme otentikasi khusus yang dirancang untuk layanan tertentu. AWS Kredensi ini menyediakan otentikasi yang disederhanakan dibandingkan dengan AWS kredensil standar, dan disesuaikan dengan persyaratan otentikasi layanan individual. AWS Tidak

seperti kunci akses, yang dapat digunakan di beberapa AWS layanan, kredensil khusus layanan dirancang untuk digunakan hanya dengan layanan yang dibuatnya. Pendekatan yang ditargetkan ini meningkatkan keamanan dengan membatasi ruang lingkup kredensialnya.

Kredensi khusus layanan biasanya terdiri dari pasangan nama pengguna dan kata sandi atau kunci API khusus yang diformat sesuai dengan persyaratan layanan tertentu. Saat Anda membuat kredensil khusus layanan, kredensialnya aktif secara default dan dapat segera digunakan. Anda dapat memiliki maksimal dua set kredensi khusus layanan untuk setiap layanan yang didukung per pengguna IAM. Batas ini memungkinkan Anda mempertahankan satu set aktif sambil memutar ke set baru bila diperlukan. AWS saat ini mendukung kredensil khusus layanan untuk layanan berikut:

Kapan menggunakan kredensil khusus layanan

Kredensil khusus layanan dimaksudkan untuk kompatibilitas dengan pustaka pihak ketiga,, alat SDKs, atau aplikasi yang tidak kompatibel secara asli dengan kredensil,, atau. AWS AWS SDKs AWS APIs Kasus penggunaan tersebut termasuk migrasi ke AWS layanan dari infrastruktur yang dihosting sendiri atau dari layanan yang dihosting oleh penyedia lain.

Saat memulai dari awal, dan sedapat mungkin, sebaiknya Anda menggunakan kredensil AWS sementara, seperti yang disediakan oleh peran IAM, untuk mengautentikasi dengan AWS layanan menggunakan AWS SDK atau pustaka yang mendukung kredensi sementara. AWS

Memutar kredensil khusus layanan

Sebagai praktik terbaik keamanan, putar kredensil khusus layanan secara teratur. Untuk memutar kredensil tanpa mengganggu aplikasi Anda:

1. Buat set kedua kredensi khusus layanan untuk layanan dan pengguna IAM yang sama
2. Perbarui semua aplikasi untuk menggunakan kredensil baru dan verifikasi bahwa mereka berfungsi dengan benar
3. Ubah status kredensil asli menjadi “Tidak Aktif”
4. Pastikan semua aplikasi masih berfungsi dengan baik
5. Hapus kredensil khusus layanan yang tidak aktif saat Anda yakin kredensil tersebut tidak lagi diperlukan

Memantau kredensil khusus layanan

Anda dapat menggunakan AWS CloudTrail untuk memantau penggunaan kredensi khusus layanan di akun Anda. AWS Untuk melihat CloudTrail peristiwa yang terkait dengan penggunaan kredensi khusus layanan, tinjau CloudTrail log untuk peristiwa dari layanan tempat kredensialnya digunakan. Untuk informasi selengkapnya, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail.

Untuk keamanan tambahan, pertimbangkan untuk menyiapkan CloudWatch alarm untuk memberi tahu Anda tentang pola penggunaan kredenal tertentu yang mungkin menunjukkan akses tidak sah atau masalah keamanan lainnya. Untuk informasi selengkapnya, lihat Memantau File CloudTrail Log dengan CloudWatch Log Amazon di Panduan AWS CloudTrail Pengguna.

Topik berikut memberikan informasi tentang kredensil khusus layanan.

Topik

- Kunci API untuk AWS layanan
- Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Kunci API untuk AWS layanan

Beberapa AWS layanan mendukung kunci API untuk mengotentikasi permintaan terprogram selain kredentif IAM standar seperti kredentif keamanan sementara dan kunci akses jangka panjang. AWS menawarkan dua jenis kunci API:

- Long-term Kunci API — Long-term Kunci API dikaitkan dengan pengguna IAM dan dibuat menggunakan kredentif khusus layanan IAM. KredenSIAL ini dirancang untuk digunakan hanya dengan satu AWS layanan, meningkatkan keamanan dengan membatasi ruang lingkup kredensia. Anda dapat mengatur waktu kedaluwarsa saat kunci API jangka panjang kedaluwarsa. Untuk menghasilkan kunci API jangka panjang, Anda dapat menggunakan IAM atau konsol khusus layanan, atau API. AWS CLI AWS
- Short-term Kunci API — Kunci API jangka pendek adalah URL yang ditandatangani sebelumnya yang menggunakan Tanda T AWS angan Versi 4. Short-term Kunci API berbagi izin dan kedaluwarsa yang sama dengan kredenSIAL identitas yang menghasilkan kunci API dan berlaku hingga 12 jam atau sisa waktu sesi konsol Anda, mana saja yang lebih pendek. Anda dapat menggunakan Amazon Bedrock/Claude Platform di AWS console, Python, dan paket untuk bahasa pemrograman lain untuk menghasilkan kunci API jangka pendek. Untuk informasi selengkapnya,

lihat Mengh asilkan kunci API Amazon Bedrock untuk akses mudah ke Amazon Bedrock API di Panduan Pengguna Amazon Bedrock dan Oten tikasi di Panduan Pengguna. Claude Platform di AWS

> **Note**
>
> Long-term Kunci API memiliki risiko keamanan yang lebih tinggi dibandingkan dengan kunci API jangka pendek. Sebaiknya gunakan kunci API jangka pendek atau kredenSIAL keamanan sementara jika memungkinkan. Jika Anda menggunakan kunci API jangka panjang, sebaiknya terapkan praktik rotasi kunci reguler.

Layanan yang mendukung kunci API

Tabel berikut mencantumkan AWS layanan yang mendukung kunci API dan jenis kunci API yang didukung setiap layanan.

#	Layanan	Long-term Kunci API	Short-term Kunci API	Kebijakan terkelola dilampirkan secara otomatis	Service-specific dokumentasi
1	Amazon Bedrock	Ya	Ya	AmazonBedrockLimitedAccess	Gunakan kunci API Amazon Bedrock
2	Claude Platform di AWS	Ya	Ya	AnthropicInferenceAccess	Autentikasi
3	Amazon CloudWatch	Ya	N/A	CloudWatchAPIKeyAccess	Menyiapkan otentikasi token pembawa untuk Metrik

#	Layanan	Long-term Kunci API	Short-term Kunci API	Kebijakan terkelola dilampirkan secara otomatis	Service-specific dokumentasi
4	CloudWatch Log Amazon	Ya	N/A	CloudWatchLogsAPIKeyAccess	Menyiapkan otentikasi token pembawa

Saat Anda membuat kunci API jangka panjang untuk layanan, kebijakan terkelola AWS terkait secara otomatis dilampirkan ke pengguna IAM, memberikan akses ke operasi inti untuk layanan tersebut. Jika Anda memerlukan akses tambahan, Anda dapat mengubah izin untuk pengguna IAM. Untuk informasi tentang memodifikasi izin, lihatthe section called “Menambah atau menghapus izin identitas”.

Untuk mempelajari lebih lanjut tentang kunci API untuk layanan tertentu, lihat tautan Service-specific dokumentasi pada tabel di atas.

Prasyarat untuk kunci API jangka panjang

Sebelum Anda dapat membuat kunci API jangka panjang di konsol IAM, Anda harus memenuhi prasyarat berikut:

- Pengguna IAM untuk dikaitkan dengan kunci API jangka panjang. Untuk petunjuk tentang cara membuat pengguna IAM, lihatthe section called “Buat pengguna”.
- Anda harus memiliki izin kebijakan IAM berikut untuk mengelola kredenSIAL khusus layanan untuk pengguna IAM. Kebijakan contoh memberikan izin untuk membuat, membuat daftar, memperbarui, menghapus, dan mengatur ulang kredenSIAL khusus layanan. Ganti *username* nilai dalam elemen Resource dengan nama pengguna IAM yang akan Anda hasilkan kunci API jangka panjang untuk:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "ManageBedrockServiceSpecificCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:UpdateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ResetServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/username"
        }
    ]
}
```

Menghasilkan kunci API jangka panjang (konsol)

Untuk menghasilkan kunci API jangka panjang untuk layanan tertentu di konsol IAM

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Pengguna.
3. Pilih pengguna IAM yang ingin Anda buat kunci API jangka panjang.
4. Pilih tab Kredensial keamanan.
5. Di bagian kunci API, pilih Buat kunci API.
6. Dari daftar dropdown AWS layanan, pilih layanan yang ingin Anda autentikasi dengan kunci API.
7. Untuk kedaluwarsa kunci API, lakukan salah satu hal berikut:
   - Pilih durasi kedaluwarsa kunci API 1, 5, 30, 90, atau 365 hari.
   - Pilih Durasi kustom untuk menentukan tanggal kedaluwarsa kunci API khusus.
   - Pilih Tidak pernah kedalu warsa (tidak disarankan).
8. Pilih Buat kunci API.
9. Salin atau unduh kunci API Anda. Ini adalah satu-satunya waktu Anda dapat melihat nilai kunci API.

> **Important**
>
> Simpan kunci API Anda dengan aman. Setelah Anda menutup kotak dialog, Anda tidak dapat mengambil kunci API lagi. Jika Anda kehilangan atau lupa kunci API Anda, Anda tidak dapat mengambilnya. Sebagai gantinya, buat kunci API baru dan buat kunci lama tidak aktif.

Menghasilkan kunci API jangka panjang (AWS CLI)

Untuk menghasilkan kunci API jangka panjang menggunakan AWS CLI, gunakan langkah-langkah berikut:

1. Buat pengguna IAM yang akan digunakan dengan layanan menggunakan perintah create- user:

```
aws iam create-user \
    --user-name APIKeyUser_1
```

2. Lampirkan kebijakan AWS terkelola ke pengguna IAM menggunakan perintah attach- user-policy.

   Untuk Amazon Bedrock:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AmazonBedrockLimitedAccess
```

   Untuk Claude Platform di AWS:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AnthropicInferenceAccess
```

   Untuk Amazon CloudWatch:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAPIKeyAccess
```

   Untuk Amazon CloudWatch Log:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchLogsAPIKeyAccess
```

3. Hasilkan kunci API jangka panjang menggunakan perintah create-service-spesifik-credential.

   Untuk Amazon Bedrock:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name bedrock.amazonaws.com \
    --credential-age-days 30
```

   Untuk Claude Platform di AWS:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name aws-external-anthropic.amazonaws.com \
    --credential-age-days 30
```

   Untuk Amazon CloudWatch:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name cloudwatch.amazonaws.com \
    --credential-age-days 30
```

   Untuk Amazon CloudWatch Log:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name logs.amazonaws.com \
    --credential-age-days 30
```

> **Note**
>
> Parameter `--credential-age-days` bersifat opsional. Anda dapat menentukan nilai antara 1-36600 hari. Jika Anda menghilangkan parameter ini, kunci API tidak kedaluwarsa.

Yang dikembalikan `ServiceApiKeyValue` dalam respons adalah kunci API jangka panjang Anda untuk layanan masing-masing. Simpan `ServiceApiKeyValue` nilai dengan aman, karena Anda tidak dapat mengambilnya nanti.

Daftar kunci API jangka panjang (AWS CLI)

Untuk mencantumkan metadata kunci API jangka panjang untuk pengguna tertentu, gunakan perintah list-service-specific dengan parameter: `--user-name`

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --user-name APIKeyUser_1
```

> **Note**
>
> Ganti `bedrock.amazonaws.com` dengan nama layanan yang sesuai (misalnya, `logs.amazonaws.com` untuk Amazon CloudWatch Logs atau `aws-external-anthropic.amazonaws.com` untuk Claude Platform di AWS).

Untuk mencantumkan semua metadata kunci API jangka panjang di akun, gunakan perintah list-service-specific dengan parameter: `--all-users`

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --all-users
```

Perbarui status kunci API jangka panjang (AWS CLI)

Untuk memperbarui status kunci API jangka panjang, gunakan perintah update-service- specific credential:

```
aws iam update-service-specific-credential \
    --user-name "APIKeyUser_1" \
    --service-specific-credential-id "ACCA1234EXAMPLE1234" \
    --status Inactive|Active
```

Menghasilkan kunci API jangka panjang (AWS API)

Anda dapat menggunakan operasi IAM API berikut untuk mengelola kunci API jangka panjang untuk layanan yang didukung:

- CreateServiceSpecificCredential
- ListServiceSpecificCredentials
- UpdateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential

Short-term Kunci API (pilih layanan)

Short-term Kunci API saat ini didukung oleh layanan tertentu.

Untuk informasi tentang membuat dan menggunakan kunci API jangka pendek dengan Amazon Bedrock, lihat Menghasilkan kunci API di Panduan Pengguna Amazon Bedrock.

Untuk informasi tentang membuat dan menggunakan kunci API jangka pendek untuk Claude Platform di AWS, lihat O tentikasi di Panduan Claude Platform di AWS Pengguna.

Service-specific informasi

- Untuk informasi selengkapnya tentang menggunakan kunci API dengan Amazon Bedrock, lihat Menggunakan kunci API Amazon Bedrock di Panduan Pengguna Amazon Bedrock.
- Untuk informasi selengkapnya tentang menggunakan kunci API dengan Claude Platform di AWS, lihat O tentikasi di Panduan Claude Platform di AWS Pengguna.
- Untuk informasi selengkapnya tentang menggunakan kunci API dengan Amazon CloudWatch, lihat Men yiapkan otentikasi token pembawa untuk Metrik di Panduan CloudWatch Pengguna Amazon.
- Untuk informasi selengkapnya tentang menggunakan kunci API dengan Amazon CloudWatch Logs, lihat Men yiapkan otentikasi token pembawa di Panduan Pengguna Amazon CloudWatch Logs.

Gunakan IAM dengan Amazon Keyspaces (untuk Apache Cassandra)

Amazon Keyspaces (untuk Apache Cassandra) adalah layanan basis data yang kompatibel dengan Apache Cassandra yang dapat diskalakan, selalu tersedia, dan terkelola. Anda dapat mengakses

Amazon Keyspaces melalui Konsol Manajemen AWS, atau secara terprogram. Untuk mengakses Amazon Keyspaces secara terprogram dengan kredenal khusus layanan, Anda dapat menggunakan atau driver Cassandra sumber terbuka. `cqlsh` Kredensi khusus layanan mencakup nama pengguna dan kata sandi seperti yang digunakan Cassandra untuk otentikasi dan manajemen akses. Anda dapat memiliki maksimal dua set kredensi khusus layanan untuk setiap layanan yang didukung per pengguna.

Untuk mengakses Amazon Keyspaces secara terprogram dengan kunci AWS akses, Anda dapat menggunakan AWS SDK, AWS Command Line Interface (AWS CLI) atau driver Cassandra open-source dengan plugin SiGv4. Untuk mempelajari selengkapnya, lihat Membuat dan mengonfigurasi AWS kredensional untuk Amazon Keyspaces di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

> **Note**
>
> Jika Anda berencana untuk berinteraksi dengan Amazon Keyspaces hanya melalui konsol, Anda tidak perlu membuat kredensial khusus layanan. Untuk informasi selengkapnya, lihat Mengakses Amazon Keyspaces menggunakan konsol di Amazon Keyspaces (untuk Apache Cassandra) Panduan Pengembang.

Untuk informasi lebih lanjut tentang izin yang diperlukan untuk mengakses Amazon Keyspaces, lihat Contoh Kebijakan Berbasis Identitas Amazon Keyspaces (untuk Apache Cassandra) dalam Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).

Membuat kredensial Amazon Keyspaces (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna IAM Anda.

Untuk menghasilkan kredensial khusus layanan Amazon Keyspaces (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna lalu pilih nama pengguna yang memerlukan kredensial.
3. Di tab Kredensial Keamanan di bawah Kredensial untuk Amazon Keyspaces (untuk Apache Cassandra), pilih Buat kredensial.
4. Kredensial layanan khusus Anda sekarang tersedia. Ini adalah satu-satunya waktu untuk melihat atau mengunduh kata sandi. Anda tidak dapat memulihkannya nanti. Namun, Anda dapat

mengatur ulang kata sandi Anda kapan saja. Simpan pengguna dan kata sandi di lokasi aman, karena Anda akan membutuhkannya nanti.

Menghasilkan kredensi Amazon Keyspaces ()AWS CLI

Anda dapat menggunakan AWS CLI untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna IAM Anda.

Untuk menghasilkan kredensi khusus layanan Amazon Keyspaces ()AWS CLI

- Gunakan perintah berikut ini.
  - aws iam create-service-specific-credential

Menghasilkan kredensi Amazon Keyspaces (API)AWS

Anda dapat menggunakan AWS API untuk menghasilkan kredensi Amazon Keyspaces (untuk Apache Cassandra) untuk pengguna IAM Anda.

Untuk menghasilkan kredensi khusus layanan (API) Amazon Keyspaces AWS

- Selesaikan operasi berikut:
  - CreateServiceSpecificCredential

Temukan yang tidak terpakai AWS credentials

Untuk meningkatkan keamanan Anda Akun AWS, hapus kredensi pengguna IAM (yaitu, kata sandi dan kunci akses) yang tidak diperlukan. Misalnya, ketika pengguna meninggalkan organisasi Anda atau tidak lagi memerlukan AWS akses, temukan kredensi yang mereka gunakan dan pastikan bahwa mereka tidak lagi beroperasi. Idealnya, Anda menghapus kredensial jika tidak lagi diperlukan. Anda dapat selalu membuat ulang catatan di kemudian hari jika perlu. Sekurang-kurangnya, Anda harus mengubah kata sandi atau menonaktifkan access key sehingga pengguna sebelumnya tidak lagi memiliki akses.

Tentu saja, definisi dari tidak digunakan dapat bervariasi dan biasanya berarti kredensial yang belum digunakan dalam jangka waktu tertentu.

Menemukan kata sandi yang tidak digunakan

Anda dapat menggunakan Konsol Manajemen AWS untuk melihat informasi penggunaan kata sandi untuk pengguna Anda. Jika Anda memiliki sejumlah besar pengguna, Anda dapat menggunakan konsol untuk mengunduh laporan kredensial dengan informasi tentang kapan terakhir pengguna menggunakan kata sandi konsol mereka. Anda juga dapat mengakses informasi dari AWS CLI atau API IAM.

Untuk menemukan yang tidak digunakan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan Masuk terakhir konsol pada tabel pengguna:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Pilih kolom yang terlihat, pilih Konsol masuk terakhir.
   c. Pilih Konfirmasi untuk kembali ke daftar pengguna.
4. Kolom login terakhir Konsol menunjukkan tanggal saat pengguna terakhir kali masuk AWS melalui konsol. Anda dapat menggunakan informasi ini untuk menemukan kata sandi pengguna yang tidak masuk lebih dari periode waktu tertentu. Kolom menampilkan Tidak pernah bagi pengguna dengan kata sandi yang belum pernah masuk. Tidak ada menunjukkan pengguna tanpa kata sandi. Kata sandi yang belum digunakan baru-baru ini mungkin merupakan calon yang baik untuk ditiadakan.

> **Important**
>
> Karena masalah layanan, data penggunaan terakhir kata sandi tidak mencakup penggunaan kata sandi dari tanggal 3 Mei 2018 22.50 PDT hingga 23 Mei 2018 14.08 PDT. Hal ini memengaruhi tanggal masuk terakhir yang ditampilkan di konsol IAM dan tanggal penggunaan terakhir kata sandi di Laporan kredensial IAM, dan dikembalikan oleh GetUser Operasi API. Jika pengguna masuk selama waktu yang terpengaruh, tanggal terakhir kali menggunakan kata sandi yang dikembalikan adalah tanggal pengguna terakhir masuk sebelum 3 Mei 2018. Untuk pengguna yang masuk setelah 23 Mei 2018 14.08 PDT, tanggal penggunaan terakhir kata sandi yang dikembalikan adalah akurat.

> Jika Anda menggunakan kata sandi informasi yang terakhir digunakan untuk mengidentifikasi kredensi yang tidak digunakan untuk penghapusan, seperti menghapus pengguna yang tidak masuk dalam 90 hari terakhir, kami sarankan Anda menyesuaikan jendela evaluasi Anda untuk menyertakan tanggal setelah 23 Mei 2018. AWS Atau, jika pengguna Anda menggunakan kunci akses untuk mengakses AWS secara terprogram, Anda dapat merujuk ke kunci akses informasi yang terakhir digunakan karena akurat untuk semua tanggal.

Untuk menemukan kata sandi yang tidak digunakan dengan mengunduh laporan kredensial (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan kredensial.
3. Pilih Unduh Laporan untuk mengunduh file CSV (nilai yang dipisahkan koma) bernama `status_reports_<date>T<time>.csv`. Kolom kelima berisi `password_last_used` kolom dengan tanggal atau salah satu dari berikut:
   - N/A— Pengguna yang tidak memiliki kata sandi yang ditetapkan sama sekali.
   - no_information – Pengguna yang belum menggunakan kata sandi mereka sejak IAM mulai melacak umur kata sandi pada 20 Oktober 2014.

Untuk menemukan kata sandi yang tidak digunakan (AWS CLI)

Jalankan perintah berikut untuk menemukan kata sandi yang tidak digunakan:

- `aws iam list-users` mengembalikan daftar pengguna, masing-masing dengan nilai `PasswordLastUsed`. Jika nilai hilang, maka pengguna tidak memiliki kata sandi atau kata sandi belum digunakan karena IAM mulai melacak usia kata sandi pada 20 Oktober 2014.

Untuk menemukan kata sandi yang tidak digunakan (AWS API)

Hubungi operasi berikut untuk menemukan kata sandi yang tidak digunakan:

- `ListUsers` mengembalikan koleksi pengguna, masing-masing memiliki nilai `<PasswordLastUsed>`. Jika nilai hilang, maka pengguna tidak memiliki kata sandi atau kata sandi belum digunakan karena IAM mulai melacak usia kata sandi pada 20 Oktober 2014.

Untuk informasi tentang perintah untuk mengunduh laporan kredensial, lihat Mendapatkan laporan kredensi (AWS CLI).

Menemukan access key yang tidak digunakan

Anda dapat menggunakan Konsol Manajemen AWS untuk melihat informasi penggunaan kunci akses bagi pengguna Anda. Jika Anda memiliki sejumlah besar pengguna, Anda dapat menggunakan konsol untuk mengunduh laporan kredensial tentang kapan terakhir pengguna menggunakan access key mereka. Anda juga dapat mengakses informasi dari AWS CLI atau API IAM.

Untuk menemukan access key yang tidak digunakan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Jika perlu, tambahkan kolom Access key terakhir digunakan pada tabel pengguna:
   a. Di atas tabel di ujung kanan, pilih ikon pengaturan ( ).
   b. Di Pilih kolom yang terlihat, pilih Kunci akses terakhir digunakan.
   c. Pilih Konfirmasi untuk kembali ke daftar pengguna.
4. Kolom kunci Access terakhir digunakan menunjukkan jumlah hari sejak pengguna terakhir diakses secara AWS terprogram. Anda dapat menggunakan informasi ini untuk menemukan kata sandi pengguna yang tidak digunakan lebih dari periode waktu tertentu. Kolom menampilkan - untuk pengguna tanpa kunci akses. Access key yang belum digunakan baru-baru ini mungkin merupakan calon yang baik untuk ditiadakan.

Untuk menemukan access key yang tidak digunakan dengan mengunduh laporan kredensial (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan Kredensial.
3. Pilih Unduh Laporan untuk mengunduh file CSV (nilai yang dipisahkan koma) bernama `status_reports_<date>T<time>.csv`. Kolom 11 sampai 13 berisi tanggal terakhir digunakan, Wilayah, dan layanan informasi untuk access key 1. Kolom 16 sampai 18 berisi

informasi yang sama untuk access key 2. Nilainya adalah N/Ajika pengguna tidak memiliki kunci akses atau pengguna belum menggunakan kunci akses sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk menemukan kunci akses yang tidak digunakan (AWS CLI)

Jalankan perintah berikut untuk menemukan access key yang tidak digunakan:

- `aws iam list-access-keys` mengembalikan informasi tentang access key untuk pengguna, termasuk `AccessKeyID`.
- `aws iam get-access-key-last-used` mengambil access key ID dan hasil pengembalian yang mencakup `LastUsedDate`, `Region` yang access key-nya terakhir digunakan, dan `ServiceName` dari layanan terakhir yang diminta. Jika `LastUsedDate` hilang, maka kunci akses belum digunakan sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk menemukan kunci akses yang tidak digunakan (AWS API)

Hubungi operasi berikut untuk menemukan access key yang tidak digunakan:

- `ListAccessKeys` mengembalikan daftar dari nilai `AccessKeyID` untuk access key yang terkait dengan pengguna yang ditentukan.
- `GetAccessKeyLastUsed` mengambil access key ID dan mengembalikan kumpulan nilai. Yang disertakan adalah `LastUsedDate`, `Region` di mana access key terakhir digunakan, dan `ServiceName` dari layanan terakhir yang diminta. Jika nilai hilang, maka pengguna tidak memiliki kunci akses atau kunci akses tidak digunakan sejak IAM mulai melacak usia kunci akses pada 22 April 2015.

Untuk informasi tentang perintah untuk mengunduh laporan kredensial, lihat Mendapatkan laporan kredensi (AWS CLI).

Hasilkan laporan kredensi untuk Anda Akun AWS

Anda dapat membuat dan mengunduh laporan kredensialy ang mencantumkan semua pengguna di akun Anda dan status berbagai kredensial mereka, termasuk kata sandi, access key, dan perangkat MFA. Anda bisa mendapatkan laporan kredensi dari Konsol Manajemen AWS, AWS SDK dan Alat Baris Per intah, atau API IAM.

> **Note**
>
> Laporan kredensi IAM hanya menyertakan kreden IAM-managed SIAL berikut: kata sandi, dua kunci akses pertama per pengguna, perangkat MFA, dan X.509 sertifikat penandatanganan. Laporan ini tidak menyertakan kredenSIAL khusus layanan (seperti CodeCommit kata sandi, kunci API jangka panjang Amazon Bedrock, atau kunci API jangka panjang Amazon CloudWatch Logs) atau kunci akses pengguna lain di luar dua yang pertama. Untuk visibilitas kredensi lengkap, gunakan ListAccessKeys API ListServiceSpecificCredentials dan.

Anda dapat menggunakan laporan kredensial untuk membantu upaya audit dan kepatuhan Anda. Anda dapat menggunakan laporan untuk mengaudit efek persyaratan siklus hidup kredensia, seperti pembaruan kata sandi dan kunci akses. Anda dapat memberikan laporan ke audit eksternal, atau memberikan izin kepada auditor sehingga dia dapat mengunduh laporan secara langsung.

Anda dapat membuat laporan kredensial sesering sekali setiap empat jam. Saat Anda meminta laporan, IAM terlebih dahulu memeriksa apakah laporan untuk laporan Akun AWS tersebut telah dibuat dalam empat jam terakhir. Jika demikian, laporan terbaru diunduh. Jika laporan terbaru untuk akun tersebut lebih lama dari empat jam, atau jika tidak ada laporan sebelumnya untuk akun tersebut, IAM akan membuat dan mengunduh laporan baru.

Topik

- Izin yang diperlukan
- Memahami format laporan
- Mendapatkan laporan kredensial (konsol)
- Mendapatkan laporan kredensi (AWS CLI)
- Mendapatkan laporan kredensi (AWS API)

Izin yang diperlukan

Izin berikut diperlukan untuk membuat dan mengunduh laporan:

- Untuk membuat laporan kredensial: `iam:GenerateCredentialReport`
- Untuk mengunduh laporan: `iam:GetCredentialReport`

Memahami format laporan

Laporan kredensial diformat sebagai file nilai yang dipisahkan oleh koma (CSV). Anda dapat membuka file CSV dengan perangkat lunak spreadsheet biasa untuk melakukan analisis, atau Anda dapat membuat aplikasi yang menggunakan file CSV secara terprogram dan melakukan analisis kustom.

File CSV berisi kolom berikut:

pengguna

Nama pengguna yang ramah.

arn

Amazon Resource Name (ARN) dari pengguna. Untuk informasi selengkapnya tentang ARN, lihat ARN IAM

user_creation_time

Tanggal dan waktu saat pengguna dibuat, dalam Format tanggal-waktu ISO 8601.

password_enabled

Ketika pengguna memiliki kata sandi, nilai ini adalah `TRUE`. Jika tidak, nilainya `FALSE`. Nilai ini `FALSE` untuk akun anggota baru yang dibuat sebagai bagian dari organisasi Anda karena mereka tidak memiliki kredentif pengguna root secara default.

password_last_used

Tanggal dan waktu ketika kata sandi Pengguna root akun AWS atau pengguna terakhir digunakan untuk masuk ke AWS situs web, dalam format tanggal-waktu ISO 8601. AWS situs web yang menangkap waktu masuk terakhir pengguna adalah Forum Konsol Manajemen AWS AWS Diskusi, dan AWS Marketplace. Ketika kata sandi digunakan lebih dari sekali dalam rentang 5 menit, hanya penggunaan pertama yang dicatat dalam bidang ini.

- Nilai dalam kolom ini adalah `no_information` dalam kasus ini:
  - Kata sandi pengguna belum pernah digunakan.
  - Tidak ada data masuk yang terkait dengan kata sandi, seperti saat kata sandi pengguna belum digunakan setelah IAM mulai melacak informasi ini pada tanggal 20 Oktober 2014.
- Nilai dalam kolom ini adalah `N/A` (tidak berlaku) jika pengguna tidak memiliki kata sandi.

> **Important**
>
> Karena masalah layanan, data penggunaan terakhir kata sandi tidak mencakup penggunaan kata sandi dari tanggal 3 Mei 2018 22.50 PDT hingga 23 Mei 2018 14.08 PDT. Hal ini memengaruhi tanggal masuk terakhir yang ditampilkan di konsol IAM dan tanggal penggunaan terakhir kata sandi di Laporan kredensial IAM, dan dikembalikan oleh GetUser Operasi API. Jika pengguna masuk selama waktu yang terpengaruh, tanggal terakhir kali menggunakan kata sandi yang dikembalikan adalah tanggal pengguna terakhir masuk sebelum 3 Mei 2018. Untuk pengguna yang masuk setelah 23 Mei 2018 14.08 PDT, tanggal penggunaan terakhir kata sandi yang dikembalikan adalah akurat.
> Jika Anda menggunakan informasi kata sandi yang terakhir digunakan untuk mengidentifikasi kredenSIAL yang tidak digunakan untuk dihapus, seperti menghapus pengguna yang tidak masuk AWS dalam 90 hari terakhir, sebaiknya sesuaikan jendela evaluasi untuk menyertakan tanggal setelah 23 Mei 2018. Atau, jika pengguna Anda menggunakan kunci akses untuk mengakses secara AWS terprogram, Anda dapat merujuk ke informasi kunci akses yang terakhir digunakan karena akurat untuk semua tanggal.

password_last_changed

Tanggal dan waktu saat kata sandi pengguna terakhir diatur, di Format tanggal-waktu ISO 8601. Jika pengguna tidak memiliki kata sandi, nilai dalam kolom ini adalah `N/A` (tidak berlaku).

password_next_rotation

Saat akun memiliki kebijakan kata sandi yang memerlukan rotasi kata sandi, kolom ini berisi tanggal dan waktu, dalam Format tanggal-waktu ISO 8601, saat pengguna diminta untuk mengatur kata sandi baru. Nilai untuk Akun AWS (root) selalu`not_supported`.

mfa_active

Saat perangkat autentikasi multi-factor (MFA) telah diaktifkan untuk pengguna, nilai ini adalah `TRUE`. Jika tidak, nilainya `FALSE`.

access_key_1_active

Saat pengguna memiliki access key dan status access key `Active`, nilainya `TRUE` Jika tidak, nilainya `FALSE`. Berlaku untuk pengguna root akun dan pengguna IAM.

access_key_1_last_rotated

Tanggal dan waktu, dalam Format tanggal-waktu ISO 8601, saat access key pengguna dibuat atau diubah terakhir. Jika pengguna tidak memiliki access key aktif, nilai dalam kolom ini adalah `N/A` (tidak berlaku). Berlaku untuk pengguna root akun dan pengguna IAM.

akses_key_1_last_used_date

Tanggal dan waktu, dalam Format tanggal-waktu ISO 8601, saat access key pengguna terakhir kali digunakan untuk menandatangani permintaan API AWS . Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.
- Kunci akses belum pernah digunakan.
- Kunci akses ini belum digunakan setelah IAM mulai melacak informasi ini pada 22 April 2015.

akses_key_1_last_used_region

Wilayah AWS yang access key-nya baru-baru ini digunakan. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.
- Kunci akses belum pernah digunakan.
- Kunci akses terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.
- Layanan terakhir yang digunakan tidak Region-specific, seperti Amazon S3.

akses_key_1_last_used_service

AWS Layanan yang terakhir diakses dengan kunci akses. Nilai dalam bidang ini menggunakan ruangnama layanan—misalnya, `s3` untuk Amazon S3 dan `ec2` untuk Amazon EC2. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key.

- Kunci akses belum pernah digunakan.
- Kunci akses terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

access_key_2_active

Saat pengguna memiliki access key kedua dan status access key kedua adalah `Active`, nilainya `TRUE`. Jika tidak, nilainya `FALSE`. Berlaku untuk pengguna root akun dan pengguna IAM.

> **Note**
>
> Pengguna dapat memiliki hingga dua tombol akses, untuk mempermudah rotasi dengan memperbarui kunci terlebih dahulu dan kemudian menghapus kunci sebelumnya. Untuk informasi selengkapnya tentang memperbarui kunci akses, lihatPerbarui kunci akses.

access_key_2_last_rotated

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, saat kunci akses kedua pengguna dibuat atau terakhir diperbarui. Jika pengguna tidak memiliki access key aktif kedua, nilai dalam bidang ini adalah `N/A` (tidak berlaku). Berlaku untuk pengguna root akun dan pengguna IAM.

akses_key_2_last_used_date

Tanggal dan waktu, dalam format tanggal-waktu ISO 8601, ketika kunci akses kedua pengguna terakhir digunakan untuk menandatangani AWS permintaan API. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM.

Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Kunci akses kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

akses_key_2_last_used_region

Wilayah AWS yang access key keduanya baru-baru ini digunakan. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM. Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Kunci akses kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.
- Layanan terakhir yang digunakan tidak Region-specific, seperti Amazon S3.

access_key_2_last_used_service

AWS Layanan yang terakhir diakses dengan kunci akses kedua pengguna. Nilai dalam bidang ini menggunakan ruangnama layanan—misalnya, `s3` untuk Amazon S3 dan `ec2` untuk Amazon EC2. Ketika access key digunakan lebih dari satu kali dalam rentang 15 menit, hanya penggunaan pertama yang dicatat dalam kolom ini. Berlaku untuk pengguna root akun dan pengguna IAM. Nilai dalam kolom ini adalah `N/A` (tidak berlaku) dalam kasus ini:

- Pengguna tidak memiliki access key kedua.
- Kunci akses kedua pengguna belum pernah digunakan.
- Kunci akses kedua pengguna terakhir digunakan sebelum IAM mulai melacak informasi ini pada 22 April 2015.

cert_1_active

Ketika pengguna memiliki sertifikat X.509 penandatanganan dan status sertifikat itu adalah`Active`, nilai ini adalah`TRUE`. Jika tidak, nilainya `FALSE`.

cert_1_last_rotated

Tanggal dan waktu, dalam Format tanggal-waktu ISO 8601, saat sertifikat penandaan pengguna dibuat atau terakhir diubah. Apabila pengguna tidak memiliki sertifikat penandatanganan aktif, nilai dalam kolom ini adalah `N/A` (tidak berlaku).

cert_2_active

Ketika pengguna memiliki sertifikat X.509 penandatanganan kedua dan status sertifikat itu adalah`Active`, nilai ini adalah`TRUE`. Jika tidak, nilainya `FALSE`.

> **Note**
>
> Pengguna dapat memiliki hingga dua sertifikat X.509 penandatanganan, untuk mempermudah rotasi sertifikat.

cert_2_last_rotated

Tanggal dan waktu, dalam Format tanggal-waktu ISO 8601, saat sertifikat tanda tangan kedua pengguna dibuat atau terakhir diubah. Apabila pengguna tidak memiliki sertifikat tanda tangan aktif kedua, nilai dalam bidang ini adalah `N/A` (tidak berlaku).

info kredensi_tambahan

Bila pengguna memiliki lebih dari dua kunci akses atau sertifikat, nilai ini adalah jumlah kunci akses tambahan atau sertifikat dan tindakan yang dapat Anda gunakan untuk membuat daftar kunci akses atau sertifikat yang terkait dengan pengguna.

Mendapatkan laporan kredensial (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk mengunduh laporan kredensi sebagai file nilai yang dipisahkan koma (CSV).

Untuk mengunduh laporan kredensial (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Laporan kredensial.
3. Pilih Unduh Laporan.

Mendapatkan laporan kredensi (AWS CLI)

Untuk mengunduh laporan kredenSIAL (AWS CLI)

1. Buat laporan kredenSIAL. AWS menyimpan satu laporan. Jika ada laporan, membuat laporan kredensial akan menimpa laporan sebelumnya. `aws iam generate-credential-report`
2. Melihat laporan terakhir yang dihasilkan: `aws iam get-credential-report`

Mendapatkan laporan kredensi (AWS API)

Untuk mengunduh laporan kredenSIAL (AWS API)

1. Buat laporan kredenSIAL. AWS menyimpan satu laporan. Jika ada laporan, membuat laporan kredensial akan menimpa laporan sebelumnya. `GenerateCredentialReport`

2. Melihat laporan terakhir yang dihasilkan: `GetCredentialReport`

Kredensi IAM untuk: Kredensi CodeCommit Git, kunci SSH, dan kunci akses AWS

CodeCommit adalah layanan kontrol versi terkelola yang menampung repositori Git pribadi di cloud. AWS Untuk menggunakan CodeCommit, Anda mengkonfigurasi klien Git Anda untuk berkomunikasi dengan CodeCommit repositori. Sebagai bagian dari konfigurasi ini, Anda memberikan kredensi IAM yang CodeCommit dapat digunakan untuk mengautentikasi Anda. IAM mendukung CodeCommit dengan tiga jenis kredensil:

- Kredensi Git, nama pengguna dan pasangan kata sandi yang dihasilkan oleh IAM yang dapat Anda gunakan untuk berkomunikasi dengan CodeCommit repositori melalui HTTPS.
- Kunci SSH, sebuah key pair public-private yang dihasilkan secara lokal yang dapat Anda kaitkan dengan pengguna IAM Anda untuk berkomunikasi CodeCommit dengan repositori melalui SSH.
- AWS kunci akses, yang dapat Anda gunakan dengan pembantu kredenal yang disertakan dengan AWS CLI untuk berkomunikasi dengan CodeCommit repositori melalui HTTPS.

> **Note**
>
> Anda tidak dapat menggunakan kunci SSH atau kredensial Git untuk mengakses repositori di lain akun AWS . Untuk mempelajari cara mengonfigurasi akses ke CodeCommit repositori untuk pengguna dan grup IAM di grup lain Akun AWS, lihat Mengonfigurasi akses lintas akun ke AWS CodeCommit repositori menggunakan peran di Panduan Pengguna.AWS CodeCommit

Lihat bagian berikut untuk informasi selengkapnya tentang setiap opsi.

Gunakan kredensi Git dan HTTPS dengan CodeCommit (disarankan)

Dengan kredensial Git, Anda membuat nama pengguna statis dan pasangan kata sandi untuk pengguna IAM Anda, kemudian menggunakan kredensial tersebut untuk koneksi HTTPS. Anda juga dapat menggunakan kredensial ini dengan alat pihak ketiga atau lingkungan pengembangan terpadu (IDE) yang mendukung kredensial Git statis.

Karena kredensial ini bersifat universal untuk semua sistem operasi yang didukung dan sistem manajemen kredensial yang paling kompatibel, lingkungan pengembangan, dan alat pengembangan perangkat lunak lainnya, ini adalah metode yang direkomendasikan. Anda dapat mengatur ulang kata sandi untuk kredensial Git kapan saja. Anda juga dapat membuat kredensial tidak aktif atau menghapusnya jika Anda tidak lagi membutuhkannya.

> **Note**
>
> Anda tidak dapat memilih nama pengguna atau kata sandi Anda sendiri untuk kredensial Git. IAM menghasilkan kredensi ini bagi Anda untuk membantu memastikan mereka memenuhi standar keamanan AWS dan mengamankan repositori di. CodeCommit Anda dapat mengunduh kredensialnya hanya sekali, pada saat mereka dihasilkan. Pastikan Anda menyimpan kredensial di lokasi yang aman. Jika perlu, Anda dapat mengatur ulang kata sandi kapan saja, tetapi melakukan hal tersebut dapat membatalkan koneksi yang dikonfigurasi dengan kata sandi lama. Anda harus mengonfigurasi ulang koneksi agar dapat menggunakan kata sandi baru sebelum terhubung.

Lihat topik berikut untuk informasi selengkapnya:

- Untuk membuat pengguna IAM, lihat Buat pengguna IAM di Akun AWS.
- Untuk membuat dan menggunakan kredensil Git CodeCommit, lihat Untuk Pengguna HTTPS yang Menggunakan Kredensial Git di Panduan Pengguna.AWS CodeCommit

> **Note**
>
> Mengubah nama pengguna IAM setelah membuat kredensial Git tidak mengubah nama pengguna kredensial Git. Nama pengguna dan kata sandi tetap sama dan masih valid.

Untuk memperbarui kredensi khusus layanan

1. Buat kredensial spesifik layanan kedua yang diatur sebagai tambahan yang saat ini digunakan.
2. Perbarui semua aplikasi Anda untuk menggunakan set kredensial baru dan validasi bahwa aplikasi berfungsi dengan baik.
3. Ubah status kredensial asli menjadi “Inactive”.
4. Pastikan semua aplikasi Anda masih bekerja.

5. Hapus kredensial khusus layanan tidak aktif.

Gunakan tombol SSH dan SSH dengan CodeCommit

Dengan koneksi SSH, Anda membuat file kunci publik dan pribadi di mesin lokal Anda yang Git dan CodeCommit gunakan untuk otentikasi SSH. Anda mengaitkan kunci publik dengan pengguna IAM Anda dan menyimpan kunci pribadi di mesin lokal Anda. Lihat topik berikut untuk informasi selengkapnya:

- Untuk membuat pengguna IAM, lihat Buat pengguna IAM di Akun AWS.
- Untuk membuat kunci publik SSH dan mengaitkannya dengan pengguna IAM, lihat Untuk Koneksi SSH di Linux, macOS, atau Unix atau lihat Untuk Koneksi SSH di Windows di Panduan Pengguna.AWS CodeCommit

> **Note**
>
> Kunci publik harus dikodekan dalam format ssh-rsa atau PEM. Panjang-bit minimum kunci publik adalah 2048 bit, dan panjang maksimum adalah 16384 bit. Ini terpisah dari ukuran file yang Anda unggah. Misalnya, Anda dapat membuat kunci 2048-bit, dan file PEM yang dihasilkan panjangnya 1679 byte. Jika Anda memberikan kunci publik Anda dalam format atau ukuran lain, Anda akan melihat pesan kesalahan yang menyatakan bahwa format kunci tidak valid.

Gunakan HTTPS dengan pembantu AWS CLI kredenal dan CodeCommit

Sebagai alternatif untuk koneksi HTTPS dengan kredensi Git, Anda dapat mengizinkan Git untuk menggunakan versi kredensi pengguna IAM Anda yang ditandatangani secara kriptografis atau peran instans Amazon EC2 kapan pun Git perlu mengautentikasi untuk berinteraksi dengan repositori. AWS CodeCommit Ini adalah satu-satunya metode koneksi untuk CodeCommit repositori yang tidak memerlukan pengguna IAM. Ini juga merupakan satu-satunya metode yang bekerja dengan akses federasi dan kredensial sementara. Lihat topik berikut untuk informasi selengkapnya:

- Untuk mempelajari lebih banyak membahas akses federasi, lihat Penyedia identitas dan federasi ke AWS dan Akses ke pengguna yang diautentikasi secara eksternal (federasi identitas).
- Untuk mempelajari lebih banyak tentang kredensial sementara, lihat Kredensial keamanan sementara di IAM dan Akses Sementara ke CodeCommit Repositori.

Pembantu AWS CLI kredenal tidak kompatibel dengan sistem pembantu kredenal lainnya, seperti Akses Rantai Kunci atau Manajemen Kredenal Windows. Ada pertimbangan konfigurasi tambahan ketika Anda mengonfigurasi koneksi HTTPS dengan pembantu kredensial. Untuk informasi selengkapnya, lihat Untuk Koneksi HTTPS di Linux, macOS, atau Unix dengan AWS CLI Credential Helper atau Koneksi HTTPS di Windows dengan AWS CLI Credential Helper di Panduan Pengguna.AWS CodeCommit

Mengelola sertifikat server di IAM

Untuk mengaktifkan koneksi HTTPS ke situs web atau aplikasi Anda AWS, Anda memerlukan sertifikat SSL/TLS server. Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), sebaiknya gunakan ACM untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Di Wilayah yang tidak didukung, Anda harus menggunakan IAM sebagai manajer sertifikat. Untuk mempelajari Wilayah mana yang didukung ACM, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

> **Important**
>
> ACM adalah alat pilihan untuk menyediakan, mengelola, dan menerapkan sertifikat server Anda. Dengan ACM Anda dapat meminta sertifikat atau menerapkan ACM atau sertifikat eksternal yang ada ke AWS sumber daya. Sertifikat yang disediakan oleh ACM gratis dan diperbarui secara otomatis. Di Wilayah yang Didukung, Anda dapat menggunakan ACM untuk mengelola sertifikat server dari konsol atau terprogram. Untuk informasi selengkapnya tentang menggunakan ACM, lihat Panduan AWS Certificate Manager Pengguna. Untuk informasi selengkapnya tentang meminta sertifikat ACM, lihat Mem inta Sertifikat Publik atau Meminta Sertifikat Pri badi di Panduan AWS Certificate Manager Pengguna. Untuk informasi tentang mengimpor sertifikat pihak ke tiga ke ACM, lihat Mengimpor Sertifikat di Panduan Pengguna AWS Certificate Manager .

Gunakan IAM sebagai manajer sertifikat hanya ketika Anda harus mendukung koneksi HTTPS di Wilayah yang tidak didukung oleh ACM. IAM mengenkripsi kunci pribadi Anda dengan aman dan menyimpan versi terenkripsinya dalam penyimpanan sertifikat IAM SSL. IAM mendukung penerapan sertifikat server di semua Wilayah, tetapi Anda harus mendapatkan sertifikat dari penyedia eksternal untuk digunakan dengan. AWS Anda tidak dapat mengunggah sertifikat ACM ke IAM. Sebagai tambahan, Anda tidak dapat mengelola sertifikat Anda dari Konsol IAM.

Untuk informasi selengkapnya tentang mengunggah sertifikat pihak ketiga ke IAM, lihat topik berikut.

Topik

- Unggah sertifikat server (AWS API)
- AWS Operasi API untuk sertifikat server
- Memecahkan masalah sertifikat server

Unggah sertifikat server (AWS API)

Untuk mengunggah sertifikat server ke IAM, Anda harus memberikan sertifikat yang sesuai dengan kunci pribadi. Jika sertifikat tidak ditandatangani sendiri, Anda juga harus memberikan rantai sertifikat. (Anda tidak memerlukan rantai sertifikat saat mengunggah sertifikat yang ditandatangani sendiri.) Sebelum Anda mengunggah sertifikat, pastikan bahwa Anda memiliki semua item ini dan bahwa mereka memenuhi kriteria berikut:

- Sertifikat harus valid pada saat diunggah. Anda tidak dapat mengunggah sertifikat sebelum periode validitas dimulai (tanggal `NotBefore` sertifikat) atau setelah kedaluwarsa (tanggal `NotAfter` sertifikat).
- Kunci pribadi harus tidak dienkripsi. Anda tidak dapat mengunggah kunci pribadi yang dilindungi oleh kata sandi atau frasa sandi. Untuk membantu mendekripsi kunci pribadi terenkripsi, lihat Memecahkan masalah sertifikat server.
- Sertifikat, kunci pribadi, dan rantai sertifikat semuanya harus ada PEM-encoded. Untuk membantu mengonversi item ini ke format PEM, lihat Memecahkan masalah sertifikat server.

Untuk menggunakan API IAM untuk mengunggah sertifikat, kirim permintaan UploadServerCertificate. Contoh berikut ini menunjukkan cara melakukan dengan AWS Command Line Interface (AWS CLI). Contoh tersebut mengasumsikan sebagai berikut:

- PEM-encoded Sertifikat disimpan dalam file bernama`Certificate.pem`.
- Ran PEM-encoded tai-rantai sertifikat disimpan dalam file bernama`CertificateChain.pem`.
- K PEM-encoded unci pribadi yang tidak dienkripsi disimpan dalam file bernama`PrivateKey.pem`.
- (Opsional) Anda ingin menandai sertifikat server dengan pasangan kunci-nilai. Misalnya, Anda dapat menambahkan tombol tanda `Department` dan nilai tanda`Engineering` untuk membantu Anda mengidentifikasi dan mengatur sertifikat Anda.

Untuk menggunakan perintah contoh berikut, ganti nama file berikut dengan nama Anda sendiri. Ganti *`ExampleCertificate`* dengan nama untuk sertifikat yang Anda unggah. Jika Anda ingin

menandai sertifikat, ganti *ExampleKey* dan *ExampleValue* tag pasangan kunci-nilai dengan nilai Anda sendiri. Ketik perintah pada satu garis kontinu. Contoh berikut mencakup jeda baris dan ruang tambahan untuk memudahkan Anda membaca.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Ketika perintah sebelumnya berhasil, perintah mengembalikan metadata tentang sertifikat yang diunggah, termasuk Amazon Resource Name (ARN), nama ramah, pengenalnya (ID), tanggal kedaluwarsanya, tanda, dan banyak lagi.

> **Note**
>
> Jika Anda mengunggah sertifikat server untuk digunakan dengan Amazon CloudFront, Anda harus menentukan jalur menggunakan `--path` opsi. Jalur harus dimulai dengan `/cloudfront` dan harus menyertakan garis miring yang menelusur (misalnya, `/cloudfront/test/`).

Untuk menggunakan AWS Tools for Windows PowerShell untuk mengunggah sertifikat, gunakan Publish-IAMServerCertificate.

AWS Operasi API untuk sertifikat server

Gunakan perintah berikut untuk melihat, menandai, mengganti nama, dan menghapus sertifikat server.

- Gunakan GetServerCertificate untuk mengambil sertifikat. Permintaan ini mengembalikan sertifikat, rantai sertifikat (jika diunggah), dan metadata tentang sertifikat.

  > **Note**
  >
  > Anda tidak dapat mengunduh atau mengambil kunci pribadi dari IAM setelah Anda mengunggahnya.

- Gunakan Get-IAMServerCertificate untuk mengambil sertifikat.
- Gunakan ListServerCertificates untuk mencantumkan sertifikat server yang Anda unggah. Permintaan mengembalikan daftar yang berisi metadata tentang setiap sertifikat.
- Gunakan Get-IAMServerCertificates untuk mencantumkan sertifikat server yang Anda unggah.
- Gunakan TagServerCertificate untuk menandai sertifikat server yang ada.
- Gunakan UntagServerCertificate untuk menghapus tag sertifikat server.
- Gunakan UpdateServerCertificate untuk mengganti nama sertifikat server atau memperbarui jalurnya.

  Contoh berikut ini menunjukkan cara melakukan ini dengan AWS CLI.

  Untuk menggunakan perintah contoh berikut, ganti nama sertifikat lama dan baru serta jalur sertifikat, lalu ketik perintah pada satu baris berkelanjutan. Contoh berikut mencakup jeda baris dan ruang tambahan untuk memudahkan Anda membaca.

  ```
  aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                    --new-server-certificate-
  name CloudFrontCertificate
                                    --new-path /cloudfront/
  ```

  Untuk menggunakan AWS Tools for Windows PowerShell untuk mengganti nama sertifikat server atau memperbarui jalurnya, gunakan Update-IAMServerCertificate.
- Gunakan DeleteServerCertificate untuk menghapus sertifikat server.

  Untuk menggunakan AWS Tools for Windows PowerShell untuk menghapus sertifikat server, gunakan Remove-IAMServerCertificate.

Memecahkan masalah sertifikat server

Sebelum Anda dapat mengunggah sertifikat ke IAM, Anda harus memastikan bahwa sertifikat, kunci pribadi, dan rantai sertifikat semuanya PEM-encoded. Anda juga harus memastikan bahwa kunci pribadi tidak terenkripsi. Lihat contoh-contoh berikut.

Example Contoh PEM-encoded sertifikat

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
```

```
-----END CERTIFICATE-----
```

Example Contoh PEM-encoded, kunci pribadi yang tidak dienkripsi

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Contoh rantai PEM-encoded sertifikat

Rantai sertifikat berisi satu atau beberapa sertifikat. Anda dapat menggunakan editor teks, perintah penyalinan di Windows, atau perintah cat Linux untuk menggabungkan file sertifikat Anda ke dalam rantai. Jika Anda menyertakan beberapa sertifikat, setiap sertifikat harus mensertifikasi sertifikat sebelumnya. Anda mencapai ini dengan menggabungkan sertifikat, termasuk sertifikat CA akar terakhir.

Contoh berikut berisi tiga sertifikat, tetapi rantai sertifikat Anda mungkin memuat lebih banyak atau lebih sedikit sertifikat.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Jika item ini tidak dalam format yang tepat untuk diunggah ke IAM, Anda dapat menggunakan OpenSSL untuk mengonversinya ke format yang tepat.

Untuk mengonversikan sertifikat atau rantai sertifikat dari DER ke PEM

1. Gunakan perintah OpenSSL x509, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *Certificate.der* dengan nama file yang berisi DER-encoded sertifikat Anda. Ganti *Certificate.pem* dengan nama file output yang disukai untuk berisi PEM-encoded sertifikat.

   ```
   openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
   ```

Untuk mengonversi kunci pribadi dari DER ke PEM

Gunakan perintah OpenSSL rsa, seperti pada contoh berikut. Dalam contoh perintah berikut, ganti *PrivateKey.der* dengan nama file yang berisi kunci DER-encoded pribadi Anda. Ganti *PrivateKey.pem* dengan nama pilihan file output untuk berisi kunci PEM-encoded pribadi.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Untuk mendekripsi kunci pribadi terenkripsi (hapus kata sandi atau frasa sandi)

Gunakan perintah OpenSSL rsa, seperti pada contoh berikut. Untuk menggunakan perintah contoh berikut, ganti *EncryptedPrivateKey.pem* dengan nama file yang berisi kunci pribadi Anda yang dienkripsi. Ganti *PrivateKey.pem* dengan nama pilihan file output untuk berisi kunci pribadi yang PEM-encoded tidak dienkripsi.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Untuk mengonversi bundel sertifikat dari PKCS#12 (PFX) ke PEM

Gunakan perintah OpenSSL pkcs12, seperti pada contoh berikut. Di perintah contoh berikut, ganti *CertificateBundle.p12* dengan nama file yang berisi bundel sertifikat yang dienkode PKCS#12 Anda. Ganti *CertificateBundle.pem* dengan nama file output yang disukai untuk berisi bundel PEM-encoded sertifikat.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Untuk mengonversi bundel sertifikat dari PKCS#7 ke PEM

Gunakan perintah OpenSSL pkcs7, seperti pada contoh berikut. Di perintah contoh berikut, ganti *CertificateBundle.p7b* dengan nama file yang berisi bundel sertifikat yang dienkode PKCS#7 Anda. Ganti *CertificateBundle.pem* dengan nama file output yang disukai untuk berisi bundel PEM-encoded sertifikat.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Grup pengguna IAM

Grup pengguna IAM adalah kumpulan pengguna IAM. Grup pengguna memungkinkan Anda menentukan izin untuk beberapa pengguna, yang dapat memudahkan pengelolaan izin bagi pengguna tersebut. Misalnya, Anda dapat memiliki grup pengguna yang disebut Admins dan memberikan izin administrator khas grup pengguna tersebut. Setiap pengguna dalam grup pengguna tersebut secara otomatis memiliki izin grup Admin. Jika pengguna baru bergabung dengan organisasi Anda dan memerlukan hak administrator, Anda dapat menetapkan izin yang sesuai dengan menambahkan pengguna ke grup pengguna Admin. Jika seseorang mengubah pekerjaan di organisasi Anda, alih-alih mengedit izin pengguna tersebut, Anda dapat menghapusnya dari grup IAM lama dan menambahkannya ke grup IAM baru yang sesuai.

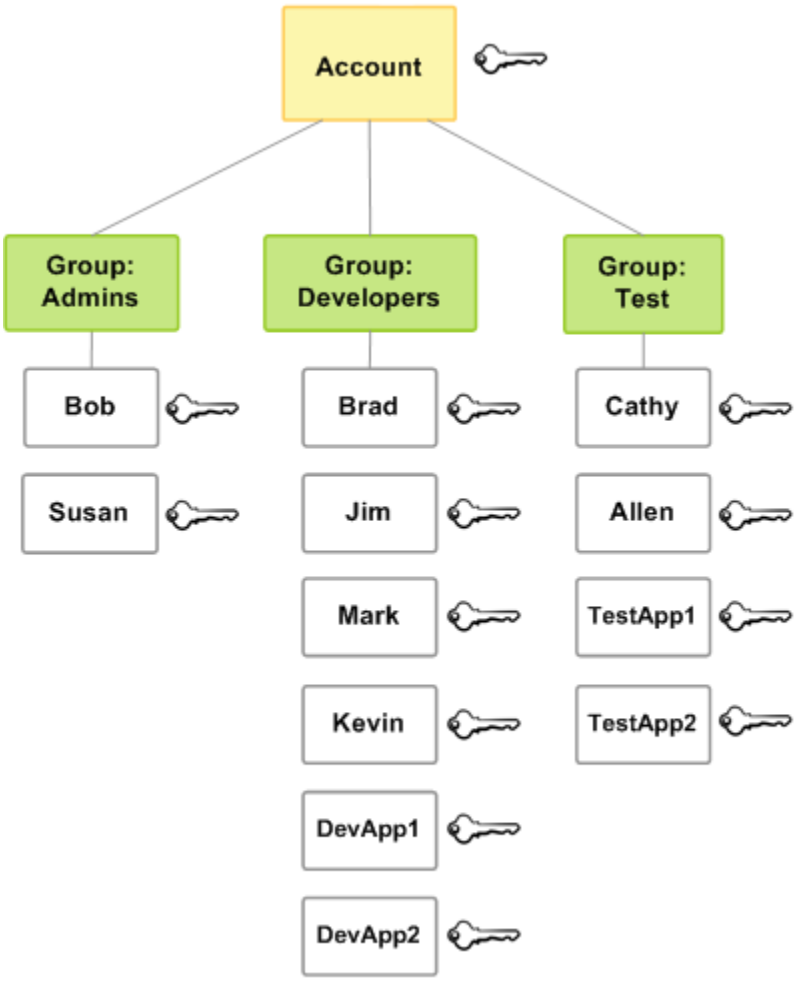
Anda dapat melampirkan kebijakan berbasis identitas ke grup pengguna sehingga semua pengguna dalam grup pengguna menerima izin kebijakan tersebut. Anda tidak dapat mengidentifikasi grup pengguna sebagai grup `Principal` dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup berhubungan dengan izin, bukan otentikasi, dan prinsipal adalah entitas IAM yang diautentikasi. Untuk informasi lebih lanjut tentang jenis kebijakan, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya.

Berikut adalah beberapa karakteristik penting dari kelompok IAM:

- Sebuah grup pengguna dapat berisi banyak pengguna, dan pengguna dapat menjadi bagian dari beberapa grup pengguna.
- Grup pengguna tidak dapat bersarang; mereka hanya dapat berisi pengguna, bukan grup IAM lainnya.
- Tidak ada grup pengguna default yang secara otomatis menyertakan semua pengguna di Akun AWS. Jika Anda ingin memiliki grup pengguna seperti itu, Anda harus membuatnya dan menetapkan setiap pengguna baru ke grup tersebut.
- Jumlah dan ukuran sumber daya IAM dalam suatu Akun AWS, seperti jumlah grup, dan jumlah grup yang dapat menjadi anggota pengguna, terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Diagram berikut menunjukkan contoh sederhana dari perusahaan kecil. Pemilik perusahaan membuat grup penggna `Admins` bagi pengguna untuk membuat dan mengelola pengguna lain

seiring pertumbuhan perusahaan. Parametergrup pengguna `Admins` membuat grup pengguna `Developers` dan grup pengguna `Test`. Masing-masing grup IAM ini terdiri dari pengguna (manusia dan aplikasi) yang berinteraksi dengan AWS (Jim, Brad, DevApp 1, dan seterusnya). Setiap pengguna memiliki satu set kredensial keamanan. Dalam contoh ini, setiap pengguna termasuk dalam satu grup pengguna. Namun, pengguna dapat menjadi bagian dari beberapa grup IAM.



Setelah Anda membuat grup pengguna, Anda dapatLihat grup IAM,Melampirkan kebijakan ke grup pengguna IAM, danGanti nama grup pengguna IAM.

Buat grup IAM

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda meminta pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan

kredensi sementara. Jika Anda mengikuti praktik terbaik, Anda tidak mengelola pengguna dan grup IAM. Sebaliknya, pengguna dan grup Anda dikelola di luar AWS dan dapat mengakses AWS sumber daya sebagai identitas federasi. Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses AWS layanan dengan menggunakan kredensi yang disediakan melalui sumber identitas. Identitas federasi menggunakan grup yang ditentukan oleh penyedia identitas mereka. Jika Anda menggunakan AWS IAM Identity Center, lihat Mengelola identitas di Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang membuat pengguna dan grup di Pusat Identitas IAM.

Anda membuat grup IAM untuk mengelola izin akses bagi beberapa pengguna dengan peran atau tanggung jawab serupa. Dengan melampirkan kebijakan ke grup ini, Anda dapat memberikan atau mencabut izin untuk seluruh kumpulan pengguna. Ini menyederhanakan pemeliharaan kebijakan keamanan Anda, karena perubahan yang Anda buat pada izin grup diterapkan secara otomatis ke semua anggota grup tersebut, memastikan kontrol akses yang konsisten. Setelah Anda membuat grup, berikan izin grup berdasarkan jenis pekerjaan yang Anda harapkan dilakukan oleh pengguna IAM dalam grup, lalu tambahkan pengguna IAM ke grup.

Untuk informasi tentang izin yang diperlukan untuk membuat grup IAM, lihat. Izin diperlukan untuk mengakses sumber daya IAM

Untuk membuat grup IAM dan melampirkan kebijakan

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Dalam panel navigasi, pilih User groups (Grup pengguna) lalu pilih Create New Group (Buat Grup Baru).
3. Untuk User grup name (Nama grup), ketik nama grup.

> **Note**
>
> Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota. Nama grup dapat berupa kombinasi hingga 128 huruf, digit, dan karakter ini: plus (+), sama (=), koma (,), titik (.), pada

tanda (@), garis bawah (_), dan tanda hubung (-). Nama harus unik dalam akun. Mereka tidak dibedakan berdasarkan kasus. Misalnya, Anda tidak dapat membuat grup dengan nama keduanya **`ADMINS`** dan **`admins`**.

4. Dalam daftar pengguna, pilih kotak centang untuk setiap pengguna yang ingin Anda tambahkan ke grup.
5. Dalam daftar kebijakan, centang kotak untuk setiap kebijakan yang ingin Anda terapkan pada semua anggota grup.
6. Pilih Buat grup.

AWS CLI

Jalankan perintah berikut:

- aws iam membuat grup

API

Panggil operasi berikut:

- CreateGroup

Lihat grup IAM

Anda dapat membuat daftar semua grup IAM di akun Anda, daftar pengguna dalam grup pengguna, dan daftar grup IAM milik pengguna. Jika Anda menggunakan CLI atau API, Anda dapat mencantumkan semua grup IAM dengan awalan jalur tertentu.

Console

Untuk mencantumkan semua grup IAM di akun Anda:

- Di panel navigasi, pilih Grup pengguna.

Untuk mencantumkan pengguna IAM dalam grup IAM tertentu:

- Pada panel navigasi, pilih User groups (Grup pengguna). Kemudian pilih nama grup untuk membuka halaman detail grup. Tinjau tab Pengguna untuk melihat keanggotaan grup.

Untuk mencantumkan semua grup IAM tempat pengguna berada:

- Di panel navigasi, pilih Users (Pengguna). Kemudian pilih nama pengguna untuk membuka halaman detail pengguna. Pilih tab Grup untuk melihat daftar grup tempat pengguna berada.

AWS CLI

Untuk mencantumkan semua grup IAM di akun Anda:

- grup daftar aws iam

Untuk membuat daftar pengguna dalam grup IAM tertentu:

- aws iam get-group

Untuk mencantumkan semua grup IAM tempat pengguna berada:

- aws iam list-groups-for-user

API

Untuk mencantumkan semua grup IAM di akun Anda:

- ListGroups

Untuk membuat daftar pengguna dalam grup IAM tertentu:

- GetGroup

Untuk mencantumkan semua grup IAM tempat pengguna berada:

- ListGroupsForUser

Mengedit pengguna dalam grup IAM

Gunakan grup IAM untuk menerapkan kebijakan izin yang sama di beberapa pengguna sekaligus. Anda kemudian dapat menambahkan pengguna ke atau menghapus pengguna dari grup IAM. Hal ini berguna saat orang-orang memasuki dan meninggalkan organisasi Anda.

Meninjau akses kebijakan

Sebelum Anda menghapus grup, gunakan halaman detail grup untuk meninjau anggota (pengguna IAM) grup, kebijakan yang dilampirkan ke grup pada tab Izin dan tinjau aktivitas tingkat layanan terbaru menggunakan tab Terakhir Diakses. Ini membantu mencegah penghapusan akses secara tidak sengaja dari kepala sekolah (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menambahkan pengguna IAM ke grup IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Pengguna dan kemudian pilih Tambahkan pengguna. Pilih kotak centang di samping pengguna yang ingin Anda tambahkan.
4. Pilih Tambahkan pengguna.

AWS CLI

Jalankan perintah berikut:

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

API

Panggil operasi berikut:

- `AddUserToGroup`

Menghapus pengguna IAM dari grup IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Pengguna. Pilih kotak centang di samping pengguna yang ingin Anda hapus, lalu pilih Hapus pengguna.

AWS CLI

Jalankan perintah berikut:

- `aws iam remove-user-from-group`

API

Panggil operasi berikut:

- `RemoveUserFromGroup`

Melampirkan kebijakan ke grup pengguna IAM

Anda dapat melampirkan kebijakan AWS terkelola —yaitu kebijakan pratulis yang disediakan oleh AWS—ke grup pengguna, seperti yang dijelaskan dalam langkah-langkah berikut. Untuk melampirkan kebijakan yang dikelola pelanggan—yaitu, izin khusus dengan kebijakan yang Anda buat—Anda harus membuat kebijakan terlebih dahulu. Untuk informasi tentang membuat kebijakan yang dikelola pelanggan, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.

Untuk informasi lebih lanjut tentang izin dan kebijakan, lihat Manajemen akses untuk AWS sumber daya.

Untuk melampirkan kebijakan ke grup IAM

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
3. Pilih tab Izin.
4. Pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.
5. Kebijakan yang saat ini terlampir ke grup pengguna ditampilkan dalam daftar Kebijakan izin saat ini. Dalam daftar Kebijakan izin lainnya, pilih kotak centang di samping nama kebijakan

yang akan dilampirkan. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan menurut jenis dan nama kebijakan.

6. Pilih kebijakan yang ingin dilampirkan ke grup IAM Anda dan pilih Lampirkan kebijakan.

AWS CLI

Jalankan perintah berikut:

- `aws iam attach-group-policy`

API

Panggil operasi berikut:

- `AttachGroupPolicy`

Ganti nama grup pengguna IAM

Saat Anda mengubah nama atau alur grup pengguna, hal berikut terjadi:

- Kebijakan apa pun yang terlampir pada grup pengguna akan tetap melekat ke grup dengan nama baru.
- Grup pengguna mempertahankan semua penggunanya di bawah nama baru.
- ID unik untuk grup pengguna tetap sama. Untuk informasi lebih lanjut tentang IDs yang unik, lihat Pengidentifikasi unik.

IAM tidak secara otomatis memperbarui kebijakan yang merujuk ke grup pengguna sebagai sumber daya untuk menggunakan nama baru. Oleh karena itu, Anda harus berhati-hati ketika mengubah nama grup pengguna. Sebelum mengubah nama grup pengguna, Anda harus memeriksa semua kebijakan secara manual untuk menemukan kebijakan yang menyebutkan nama grup pengguna tersebut. Misalnya, katakanlah Bob adalah manajer bagian pengujian dari organisasi. Bob memiliki kebijakan yang melekat pada pengguna IAM-nya yang memungkinkannya menambahkan dan menghapus pengguna dari grup pengguna Test. Jika administrator mengubah nama grup pengguna (atau mengubah jalur grup), administrator juga harus memperbarui kebijakan yang dilampirkan ke Bob untuk menggunakan nama atau jalur baru. Jika tidak, Bob tidak akan dapat menambahkan dan menghapus pengguna dari grup pengguna.

Untuk menemukan kebijakan yang merujuk ke grup IAM sebagai sumber daya:

1. Dari panel navigasi konsol IAM, pilih Kebijakan.
2. Urutkan menurut kolom Jenis untuk kebijakan kustom yang Dikelola pelanggan.
3. Pilih nama kebijakan yang ingin diedit.
4. Pilih tab Izin, lalu pilih Ringkasan.
5. Pilih IAM dari daftar layanan, jika ada.
6. Cari nama grup pengguna Anda di kolom Sumber Daya.
7. Pilih Edit untuk mengubah nama grup pengguna Anda dalam kebijakan.

Untuk mengubah nama grup pengguna IAM

Console

1. Di panel navigasi, pilih Grup pengguna lalu pilih nama grup.
2. Pilih Edit. Ketik nama grup pengguna baru lalu pilih Simpan perubahan.

AWS CLI

Jalankan perintah berikut:

- grup pembaruan aws iam

API

Panggil operasi berikut:

- UpdateGroup

Menghapus grup IAM

Saat Anda menghapus grup IAM di konsol, konsol secara otomatis menghapus semua anggota grup, melepaskan semua kebijakan terkelola yang dilampirkan, dan menghapus semua kebijakan sebaris. Namun, karena IAM tidak secara otomatis menghapus kebijakan yang merujuk ke grup IAM sebagai sumber daya, Anda harus berhati-hati saat menghapus grup IAM. Sebelum menghapus grup IAM, tinjau kebijakan Anda secara manual untuk menemukan kebijakan apa pun yang menyebutkan

grup berdasarkan nama. Misalnya, John, manajer Tim Uji, memiliki kebijakan yang dilampirkan ke entitas pengguna IAM-nya yang memungkinkannya menambah dan menghapus pengguna dari grup pengguna Uji. Jika administrator menghapus grup tersebut, administrator juga harus menghapus kebijakan yang terlampir pada John. Jika tidak, jika administrator membuat ulang grup yang dihapus dan memberinya nama yang sama, izin John tetap ada, bahkan jika dia meninggalkan Tim Uji.

Sebaliknya, saat Anda menggunakan CLI, SDK, atau API untuk menghapus grup pengguna, Anda menghapus pengguna di grup terlebih dahulu. Kemudian Anda menghapus kebijakan sebaris yang disematkan dalam grup IAM. Selanjutnya, Anda melepaskan kebijakan terkelola yang dilampirkan ke grup. Kemudian Anda menghapus grup IAM itu sendiri.

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih User groups (Grup pengguna).
3. Dalam daftar grup IAM, pilih kotak centang di sebelah nama grup IAM yang akan dihapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar grup IAM berdasarkan jenis, izin, dan nama grup.
4. Pilih Hapus.
5. Di kotak konfirmasi, jika Anda ingin menghapus satu grup, ketikkan nama grup dan pilih Hapus. Jika Anda ingin menghapus beberapa grup, ketikkan jumlah grup IAM yang akan dihapus diikuti **`user groups`** dan pilih Hapus. Misalnya, jika Anda ingin menghapus tiga grup, ketik**`3 user groups`**.

AWS CLI

1. Hapus semua pengguna dari grup IAM.
   - aws iam get-group (untuk mendapatkan daftar pengguna di grup IAM), dan aws iam remove-user-from-group (untuk menghapus pengguna dari grup IAM)
2. Hapus semua kebijakan sebaris yang disematkan dalam grup IAM.
   - aws iam list-group-policies (untuk mendapatkan daftar kebijakan sebaris grup IAM), dan aws iam delete-group-policy (untuk menghapus kebijakan sebaris grup IAM)
3. Lepaskan semua kebijakan terkelola yang dilampirkan pada grup IAM.

- aws iam list-attached-group-policies (untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup IAM), dan aws iam detach-group-policy (untuk melepaskan kebijakan terkelola dari grup IAM)

4. Hapus grup IAM.

   - aws iam hapus-grup

API

1. Hapus semua pengguna dari grup IAM.

   - GetGroup(untuk mendapatkan daftar pengguna di grup IAM) dan RemoveUserFromGroup(untuk menghapus pengguna dari grup IAM)

2. Hapus semua kebijakan sebaris yang disematkan dalam grup IAM.

   - ListGroupPolicies(untuk mendapatkan daftar kebijakan sebaris grup IAM) dan DeleteGroupPolicy(untuk menghapus kebijakan sebaris grup IAM)

3. Lepaskan semua kebijakan terkelola yang dilampirkan pada grup IAM.

   - ListAttachedGroupPolicies(untuk mendapatkan daftar kebijakan terkelola yang dilampirkan ke grup IAM) dan DetachGroupPolicy(untuk melepaskan kebijakan terkelola dari grup IAM)

4. Hapus grup IAM.

   - DeleteGroup

Peran IAM

Sebuah peran IAM adalah identitas IAM yang dapat Anda buat di akun yang memiliki izin tertentu. Peran IAM mirip dengan pengguna IAM, karena itu adalah AWS identitas dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan identitas. AWS Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensial jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran.

Anda dapat menggunakan peran untuk mendelegasikan akses ke pengguna, aplikasi, atau layanan yang biasanya tidak memiliki akses ke AWS sumber daya Anda. Misalnya, Anda mungkin ingin

memberi pengguna di AWS akun Anda akses ke sumber daya yang biasanya tidak mereka miliki, atau memberikan pengguna dalam satu Akun AWS akses ke sumber daya di akun lain. Atau Anda mungkin ingin mengizinkan aplikasi seluler menggunakan AWS sumber daya, tetapi tidak ingin menyematkan AWS kunci di dalam aplikasi (di mana kunci tersebut mungkin sulit untuk diperbarui dan di mana pengguna berpotensi mengekstraknya). Terkadang Anda ingin memberikan AWS akses ke pengguna yang sudah memiliki identitas yang ditentukan di luar AWS, seperti di direktori perusahaan Anda. Atau, Anda mungkin ingin memberikan akses ke akun Anda kepada pihak ketiga sehingga mereka dapat melakukan audit pada sumber daya Anda.

Untuk skenario ini, Anda dapat mendelegasikan akses ke AWS sumber daya menggunakan peran IAM. Bagian ini memperkenalkan peran dan berbagai cara yang dapat Anda gunakan, kapan dan bagaimana memilih di antara pendekatan, dan bagaimana membuat, mengelola, beralih ke (atau mengasumsikan), dan menghapus peran.

Manaj er akses akun adalah fitur IAM yang memungkinkan Anda menetapkan peran IAM secara terpusat di seluruh akun organisasi Anda kepada pengguna dan grup Pusat Identitas IAM. Manajer akses akun memberikan akses tenaga kerja Anda melalui peran IAM dengan set fitur lengkapnya. Anda dapat menggunakannya bersama kumpulan izin Pusat Identitas IAM atau sendiri.

> **Note**
>
> Saat Anda pertama kali membuat Akun AWS, tidak ada peran yang dibuat secara default. Saat Anda menambahkan layanan ke akun Anda, mereka dapat menambahkan peran terkait layanan untuk mendukung kasus penggunaan mereka.
> Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat mengambil peran untuk melakukan tindakan atas nama Anda. Service-linked peran muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.
> Sebelum Anda dapat menghapus peran terkait layanan, Anda harus menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.
> Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Topik

- Kapan harus membuat pengguna IAM (alih-alih peran)
- Istilah dan konsep peran
- Sumber daya tambahan
- Masalah confused deputy
- Skenario umum untuk peran IAM
- Pembuatan peran IAM
- Manajemen peran IAM
- Metode untuk mengambil peran

Kapan harus membuat pengguna IAM (alih-alih peran)

Kami menyarankan Anda hanya menggunakan pengguna IAM untuk kasus penggunaan yang tidak didukung oleh federasi identitas. Beberapa kasus penggunaan meliputi yang berikut:

- Beban kerja yang tidak dapat menggunakan peran IAM — Anda mungkin menjalankan beban kerja dari lokasi yang perlu diakses. AWS Dalam beberapa situasi, Anda tidak dapat menggunakan peran IAM untuk memberikan kredenSIAL sementara, seperti untuk WordPress plugin. Dalam situasi ini, gunakan kunci akses jangka panjang pengguna IAM untuk beban kerja tersebut untuk diautentikasi. AWS
- Third-party AWS klien — Jika Anda menggunakan alat yang tidak mendukung akses dengan Pusat Identitas IAM, seperti AWS klien pihak ketiga atau vendor yang tidak di-host AWS, gunakan kunci akses jangka panjang pengguna IAM.
- AWS CodeCommit akses — Jika Anda menggunakan CodeCommit untuk menyimpan kode Anda, Anda dapat menggunakan pengguna IAM dengan kunci SSH atau kredenSIAL khusus layanan untuk mengotentikasi CodeCommit ke repositori Anda. Kami menyarankan Anda melakukan ini selain menggunakan pengguna di Pusat Identitas IAM untuk otentikasi normal. Pengguna di Pusat Identitas IAM adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi Anda Akun AWS atau ke aplikasi cloud Anda. Untuk memberi pengguna akses ke CodeCommit repositori Anda tanpa mengkonfigurasi pengguna IAM, Anda dapat mengonfigurasi utilitasgit-remote-codecommit. Untuk informasi lebih lanjut tentang IAM dan CodeCommit, lihatKredensi IAM untuk: Kredensi CodeCommit Git, kunci SSH, dan kunci akses AWS. Untuk informasi selengkapnya tentang mengonfigurasi git-remote-codecommit utilitas, lihat Menghubungkan ke AWS CodeCommit repositori dengan kredenSIAL berputar di Panduan AWS CodeCommit Pengguna.

- Akses Amazon Keyspaces (untuk Apache Cassandra) — Dalam situasi di mana Anda tidak dapat menggunakan pengguna di Pusat Identitas IAM, seperti untuk tujuan pengujian kompatibilitas Cassandra, Anda dapat menggunakan pengguna IAM dengan kredenSIAL khusus layanan untuk mengotentikasi dengan Amazon Keyspaces. Pengguna di Pusat Identitas IAM adalah orang-orang di tenaga kerja Anda yang membutuhkan akses ke aplikasi Anda Akun AWS atau ke aplikasi cloud Anda. Anda juga dapat terhubung ke Amazon Keyspaces menggunakan kredenSIAL sementara. Untuk informasi selengkapnya, lihat Menggunakan kredenSIAL sementara untuk menyambung ke ruang kunci Amazon menggunakan peran IAM dan plugin SIGv4 di Panduan Pengembang Amazon Keyspaces (untuk Apache Cassandra).
- Akses darurat — Dalam situasi di mana Anda tidak dapat mengakses penyedia identitas Anda dan Anda harus mengambil tindakan dalam hal Anda Akun AWS. Membangun pengguna IAM akses darurat dapat menjadi bagian dari rencana ketahanan Anda. Kami menyarankan agar kredentif pengguna darurat dikontrol dan diamankan dengan ketat menggunakan otentikasi multi-faktor (MFA).

Istilah dan konsep peran

Berikut ini beberapa istilah dasar untuk membantu Anda memulai dengan peran.

Peran

Sebuah identitas IAM adalah yang dapat Anda buat di akun Anda yang memiliki izin spesifik. Peran IAM memiliki beberapa kesamaan dengan pengguna IAM. Peran dan pengguna adalah identitas AWS dengan kebijakan izin yang menentukan apa yang dapat dan tidak dapat dilakukan oleh identitas tersebut di AWS. Namun, alih-alih secara unik terkait dengan satu orang, peran dimaksudkan untuk menjadi dapat diambil oleh siapa pun yang membutuhkannya. Selain itu, peran tidak memiliki kredensial jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebagai gantinya, saat Anda mengambil peran, peran tersebut akan memberikan kredensial keamanan sementara untuk sesi peran.

Peran dapat diasumsikan dengan yang berikut:

- Pengguna IAM yang sama Akun AWS atau yang lain Akun AWS
- Peran IAM di akun yang sama
- Prinsip layanan, untuk digunakan dengan AWS layanan dan fitur seperti:
  - Layanan yang memungkinkan Anda menjalankan kode pada layanan komputasi, seperti Amazon EC2 atau AWS Lambda

  - Fitur yang melakukan tindakan ke sumber daya atas nama Anda, seperti replikasi objek Amazon S3
  - Layanan yang memberikan kredentif keamanan sementara ke aplikasi Anda yang berjalan di luar AWS, seperti IAM Roles Anywhere atau Amazon ECS Anywhere
- Pengguna eksternal yang diautentikasi oleh layanan penyedia identitas eksternal (IdP) yang kompatibel dengan SAML 2.0 atau OpenID Connect

AWS peran layanan

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

AWS peran terkait layanan

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat mengambil peran untuk melakukan tindakan atas nama Anda. Service-linked peran muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

> **Note**
>
> Jika Anda sudah menggunakan layanan ketika mulai mendukung peran yang terkait dengan layanan, Anda mungkin menerima email yang mengumumkan peran baru di akun Anda. Dalam hal ini, layanan secara otomatis membuat peran terkait layanan di akun Anda. Anda tidak perlu melakukan tindakan apa pun untuk mendukung peran ini, dan Anda tidak harus menghapusnya secara manual. Untuk informasi selengkapnya, lihat Peran baru muncul di AWS akun.

Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut. Untuk informasi selengkapnya, lihat Buat peran tertaut layanan.

Rantaian peran

Rantaian peran adalah ketika Anda menggunakan peran untuk mengambil peran kedua. Anda dapat melakukan rantai peran melalui Konsol Manajemen AWS dengan mengalihkan peran AWS

CLI, atau API. Misalnya, `RoleA` memiliki izin untuk berasumsi`RoleB`. Anda dapat mengaktifkan User1 untuk mengasumsikan `RoleA` dengan menggunakan kredentif pengguna jangka panjang mereka dalam operasi AssumeRole API. Ini mengembalikan kreden `RoleA` SIAL jangka pendek. Dengan rantai peran, Anda dapat menggunakan `RoleA` kredenSIAL jangka pendek untuk mengaktifkan User1 untuk mengasumsikan. `RoleB`

Saat Anda mengambil peran, Anda dapat melewatkan tag sesi dan mengatur tag sebagai transitif. Tag sesi transitif diteruskan ke semua sesi berikutnya dalam rantai peran. Untuk mempelajari lebih lanjut tentang tag sesi, lihat Lewati tag sesi di AWS STS.

Rantaian peran membatasi sesi peran Anda Konsol Manajemen AWS, AWS CLI atau AWS API hingga maksimal satu jam. Ini berlaku terlepas dari durasi sesi maksimum yang dikonfigurasi untuk peran individu. Saat Anda menggunakan operasi API AssumeRole untuk mengambil peran, Anda dapat menentukan durasi sesi peran Anda dengan parameter `DurationSeconds`. Anda dapat menentukan nilai parameter hingga 43200 detik (12 jam), tergantung pada pengaturan durasi sesi maksimum untuk peran Anda. Akan tetapi, jika Anda berperan menggunakan rantai peran dan memberikan nilai parameter `DurationSeconds` lebih dari satu jam, operasi gagal.

Untuk informasi tentang beralih ke peran di Konsol Manajemen AWS, lihatBeralih dari pengguna ke peran IAM (konsol).

Delegasi

Pemberian izin kepada seseorang untuk memungkinkan akses ke sumber daya yang Anda kontrol. Delegasi melibatkan terbentuknya kepercayaan antara dua akun. Yang pertama adalah akun yang memiliki sumber daya (akun terpercaya). Kedua adalah akun yang berisi pengguna yang perlu mengakses sumber daya (akun tepercaya). Akun tepercaya dan dipercaya dapat merupakan salah satu dari berikut ini:

- Akun yang sama.
- Pisahkan akun yang berada di bawah kendali organisasi Anda.
- Dua akun yang dimiliki oleh organisasi yang berbeda.

Untuk mendelegasikan izin untuk mengakses sumber daya, Anda membuat peran IAM di akun yang dipercaya yang memiliki dua kebijakan terlampir. Kebijakan izin memberikan izin yang diperlukan pengguna untuk melaksanakan tugas yang diinginkan pada sumber daya. Kebijakan kepercayaan menentukan anggota akun tepercaya mana yang diizinkan untuk menjalankan peran tersebut.

Saat Anda membuat kebijakan kepercayaan, Anda tidak dapat menentukan wildcard (*) sebagai bagian dari ARN di elemen utama. Kebijakan terpercaya ini dilampirkan pada peran dalam akun kepercayaan, dan merupakan setengah dari izin. Setengah lainnya adalah kebijakan izin yang melekat pada pengguna dalam akun tepercaya yang memungkinkan pengguna untuk beralih, atau mengambil peran. Seorang pengguna yang mengambil peran secara sementara memberikan izinnya sendiri dan sebaliknya mengambil izin peran tersebut. Saat pengguna keluar, atau berhenti menggunakan peran tersebut, izin pengguna yang asli akan dipulihkan. Parameter tambahan yang disebut ID eksternal membantu memastikan penggunaan peran aman antar akun yang tidak dikontrol oleh organisasi yang sama.

Kebijakan kepercayaan

Dokumen kebijakan JSON yang Anda definisikan menjadi prinsip yang Anda percaya untuk mengambil peran tersebut. Kebijakan kepercayaan adalah kebijakan berbasis sumber daya yang diperlukan yang melekat pada peran dalam IAM. Prinsipal yang dapat Anda sebutkan dalam kebijakan kepercayaan termasuk pengguna, peran, akun, dan layanan. Untuk informasi selengkapnya, lihat Cara menggunakan kebijakan kepercayaan dalam peran IAM di Blog AWS Keamanan.

Peran untuk akses lintas akun

Peran yang memberikan akses ke sumber daya dalam satu akun ke prinsipal tepercaya dalam akun yang berbeda. Peran adalah cara utama untuk memberikan akses akun silang. Namun, beberapa AWS layanan memungkinkan Anda melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Ini disebut kebijakan berbasis sumber daya, dan Anda dapat menggunakannya untuk memberikan prinsipal dalam Akun AWS akses lain ke sumber daya. Beberapa sumber daya ini termasuk bucket Amazon Simple Storage Service (S3), brankas Amazon Glacier, topik Amazon Simple Notification Service (SNS), dan antrian Amazon Simple Queue Service (SQS). Untuk mempelajari layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk informasi selengkapnya tentang kebijakan berbasis sumber daya, lihat Akses sumber daya lintas akun di IAM.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang terminologi IAM yang terkait dengan peran IAM.

- Prinsipal adalah entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Prinsipal dapat berupa Pengguna root akun AWS, pengguna IAM, atau peran. Prinsipal yang

mewakili identitas suatu AWS layanan adalah prinsi pal layanan. Gunakan elemen Principal dalam kebijakan kepercayaan peran untuk menentukan prinsipal yang Anda percayai untuk mengambil peran.

Untuk informasi lebih lanjut dan contoh prinsipal yang dapat Anda izinkan untuk mengambil peran, lihatAWS Elemen kebijakan JSON: Principal.

- Federasi identitas menciptakan hubungan kepercayaan antara penyedia identitas eksternal dan AWS. Anda dapat menggunakan penyedia OpenID Connect (OIDC) atau Security Assertion Markup Language (SAML) 2.0 yang ada untuk mengelola siapa yang dapat mengakses sumber daya. AWS Bila Anda menggunakan OIDC dan SAML 2.0 untuk mengonfigurasi hubungan kepercayaan antara penyedia identitas eksternal ini dan AWS , pengguna ditugaskan ke peran IAM. Pengguna juga menerima kredensial sementara yang mengizinkan pengguna mengakses sumber daya AWS Anda.

  Untuk informasi lebih lanjut tentang prinsipal federasi, lihat. Penyedia identitas dan federasi ke AWS
- Prinsipal federasi adalah identitas yang ada dari Directory Service, direktori pengguna perusahaan Anda, atau penyedia OIDC. AWS memberikan peran ke prinsipal federasi ketika akses diminta melalui penyedia identitas.

  Untuk informasi lebih lanjut tentang prinsipal federasi SAML dan OIDC, lihat. Sesi dan peran pengguna gabungan
- Kebijakan izin adalah kebijakan berbasis identitas yang menentukan tindakan dan sumber daya apa yang dapat digunakan peran tersebut. Dokumen ini ditulis sesuai dengan aturan Bahasa kebijakan IAM.

  Untuk informasi selengkapnya, lihat Referensi kebijakan IAM JSON.
- Batas izin adalah fitur lanjutan di mana Anda menggunakan kebijakan untuk membatasi izin maksimum yang dapat diberikan kebijakan berbasis identitas ke peran. Anda tidak dapat menerapkan batas izin untuk peran yang terkait layanan.

  Lihat informasi yang lebih lengkap di Batas izin untuk entitas IAM.

Masalah confused deputy

Masalah "confused deputy" adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memengaruhi entitas yang memiliki hak akses lebih tinggi untuk

melakukan tindakan. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi akun Anda jika Anda memberi pihak ketiga (dikenal sebagai lintas akun) atau AWS layanan lain (dikenal sebagai lintas layanan) akses ke sumber daya di akun Anda.

Terkadang, Anda mungkin perlu memberi pihak ketiga akses ke AWS sumber daya Anda (akses delegasi). Misalnya, Anda memutuskan untuk menyewa perusahaan pihak ketiga bernama Example Corp untuk memantau Akun AWS dan membantu mengoptimalkan biaya. Untuk melacak pengeluaran harian Anda, Example Corp perlu mengakses AWS sumber daya Anda. Contoh Corp juga memantau banyak lainnya Akun AWS untuk pelanggan lain. Anda dapat menggunakan peran IAM untuk membangun hubungan tepercaya antara akun Anda Akun AWS dan akun Example Corp. Salah satu aspek penting dari skenario ini adalah ID eksternal, pengidentifikasi opsional yang dapat Anda gunakan dalam kebijakan kepercayaan peran IAM untuk menentukan siapa yang dapat mengambil peran tersebut. Fungsi utama dari ID eksternal adalah mengatasi dan mencegah masalah deputi yang membingungkan.

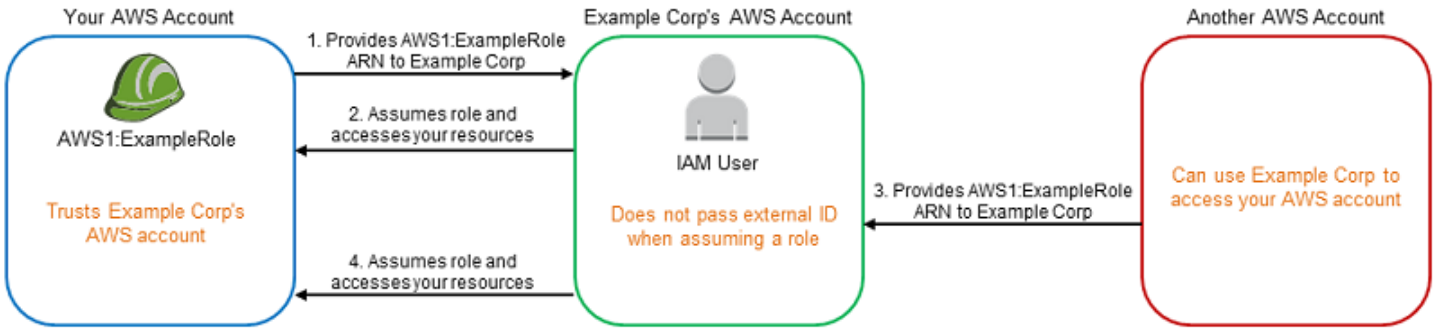
Beberapa AWS layanan (layanan panggilan) menggunakan prinsip AWS layanan mereka untuk mengakses AWS sumber daya dari AWS layanan lain (disebut layanan). Dalam beberapa interaksi layanan ini, Anda dapat mengonfigurasi layanan panggilan untuk berkomunikasi dengan sumber daya dari layanan yang disebut secara berbeda Akun AWS. Contohnya adalah mengonfigurasi AWS CloudTrail untuk menulis ke bucket Amazon S3 pusat yang terletak di bucket berbeda. Akun AWS Layanan panggilan, CloudTrail, diberikan akses ke bucket S3 Anda menggunakan kebijakan bucket S3 dengan menambahkan pernyataan allow untuk. `cloudtrail.amazonaws.com`

Ketika kepala AWS layanan dari layanan panggilan mengakses sumber daya dari layanan yang disebut, kebijakan sumber daya dari layanan yang disebut hanya mengotorisasi kepala AWS layanan, dan bukan aktor yang mengkonfigurasi layanan panggilan. Misalnya, bucket S3 yang mempercayai prinsip CloudTrail layanan tanpa kondisi dapat menerima CloudTrail log dari Akun AWS yang dikonfigurasi oleh admin tepercaya, tetapi juga CloudTrail log dari aktor yang tidak sah di dalamnya Akun AWS, jika mereka mengetahui nama bucket S3.

Masalah wakil yang membingungkan muncul ketika seorang aktor menggunakan kepercayaan dari kepala layanan untuk mendapatkan akses ke sumber daya yang tidak dimaksudkan untuk mereka akses. AWS

Cross-account pencegahan wakil bingung

Diagram berikut menggambarkan masalah wakil kebingungan lintas akun.



Skenario ini mengasumsikan hal berikut:

- AWS 1 adalah milikmu Akun AWS.
- AWS 1: ExampleRole adalah peran dalam akun Anda. Kebijakan kepercayaan peran ini mempercayai Example Corp dengan menentukan akun AWS Example Corp sebagai akun yang dapat mengambil peran tersebut.

Inilah yang terjadi:

1. Saat Anda mulai menggunakan layanan Example Corp, Anda memberikan ARN 1ExampleRole: ke AWS Example Corp.
2. Contoh Corp menggunakan peran ARN itu untuk mendapatkan kredensyal keamanan sementara untuk mengakses sumber daya di Anda. Akun AWS Dengan cara ini, Anda mempercayai Example Corp sebagai “deputi” yang dapat bertindak atas nama Anda.
3. AWS Pelanggan lain juga mulai menggunakan layanan Example Corp, dan pelanggan ini juga menyediakan ARN 1ExampleRole: untuk AWS Example Corp untuk digunakan. Agaknya pelanggan lain belajar atau menebak AWS 1: ExampleRole, yang bukan rahasia.
4. Ketika pelanggan lain meminta Example Corp untuk mengakses AWS sumber daya di (apa yang diklaimnya) akunnya, Example Corp menggunakan AWS 1: ExampleRole untuk mengakses sumber daya di akun Anda.

Inilah cara pelanggan lain bisa mendapatkan akses tanpa izin ke sumber daya Anda. Karena pelanggan lain mampu mengelabui Example Corp untuk melakukan tindakan di luar kesadaran pada sumber daya Anda, Example Corp kini adalah “confused deputy.”

Contoh Corp dapat mengatasi masalah wakil yang membingungkan dengan mengharuskan Anda memasukkan pemeriksaan `ExternalId` kondisi dalam kebijakan kepercayaan peran. Contoh Corp menghasilkan `ExternalId` nilai unik untuk setiap pelanggan dan menggunakan nilai itu dalam

permintaannya untuk mengambil peran tersebut. `ExternalId`Nilai harus unik di antara pelanggan Example Corp dan dikendalikan oleh Example Corp, bukan pelanggannya. Inilah mengapa Anda mendapatkannya dari Example Corp dan Anda tidak membuatnya sendiri. Ini mencegah Example Corp menjadi wakil yang bingung dan memberikan akses ke sumber daya akun lain. AWS

Dalam skenario kami, bayangkan pengenal unik Example Corp untuk Anda adalah 12345, dan pengenalnya untuk pelanggan lain adalah 67890. Pengidentifikasi ini disederhanakan untuk skenario ini. Umumnya, pengidentifikasi ini adalah GUID. Dengan asumsi pengidentifikasi ini bersifat unik di antara pelanggan Example Corp, mereka adalah nilai yang masuk akal untuk digunakan bagi ID eksternal.

Contoh Corp memberikan nilai ID eksternal 12345 kepada Anda. Anda kemudian harus menambahkan elemen `Condition` ke kebijakan kepercayaan peran yang memerlukan nilai `sts:ExternalId` 12345, seperti ini:

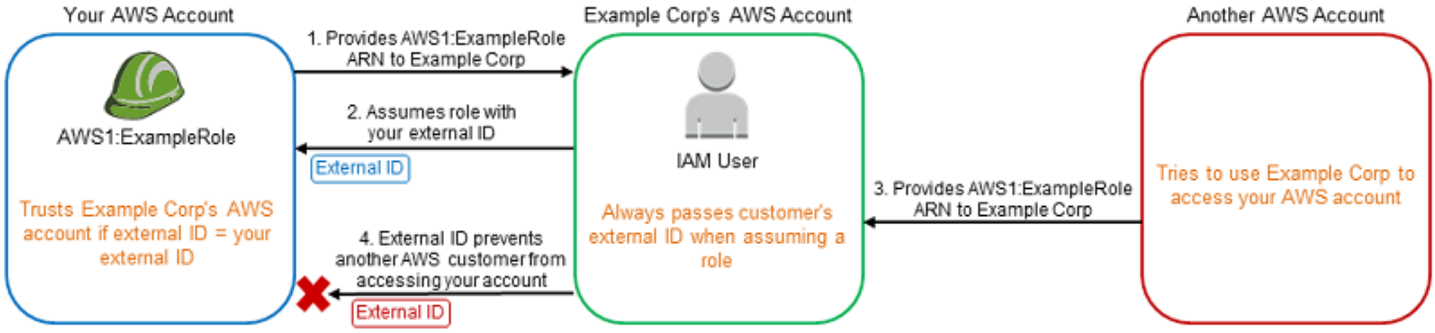
JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

Elemen Kondisi dalam kebijakan ini memungkinkan Example Corp untuk mengambil peran hanya jika panggilan AssumeRole API menyertakan nilai ID eksternal 12345. Contoh Corp memastikan bahwa setiap kali mengambil peran atas nama pelanggan, selalu menyertakan nilai ID eksternal pelanggan tersebut AssumeRole dalam panggilan. Bahkan jika pelanggan lain memasok Example Corp dengan ARN Anda, ia tidak dapat mengontrol ID eksternal yang disertakan Example Corp

dalam permintaannya. AWS Hal ini membantu mencegah pelanggan yang tidak berwenang untuk mendapatkan akses ke sumber daya Anda.

Diagram berikut menggambarkannya.

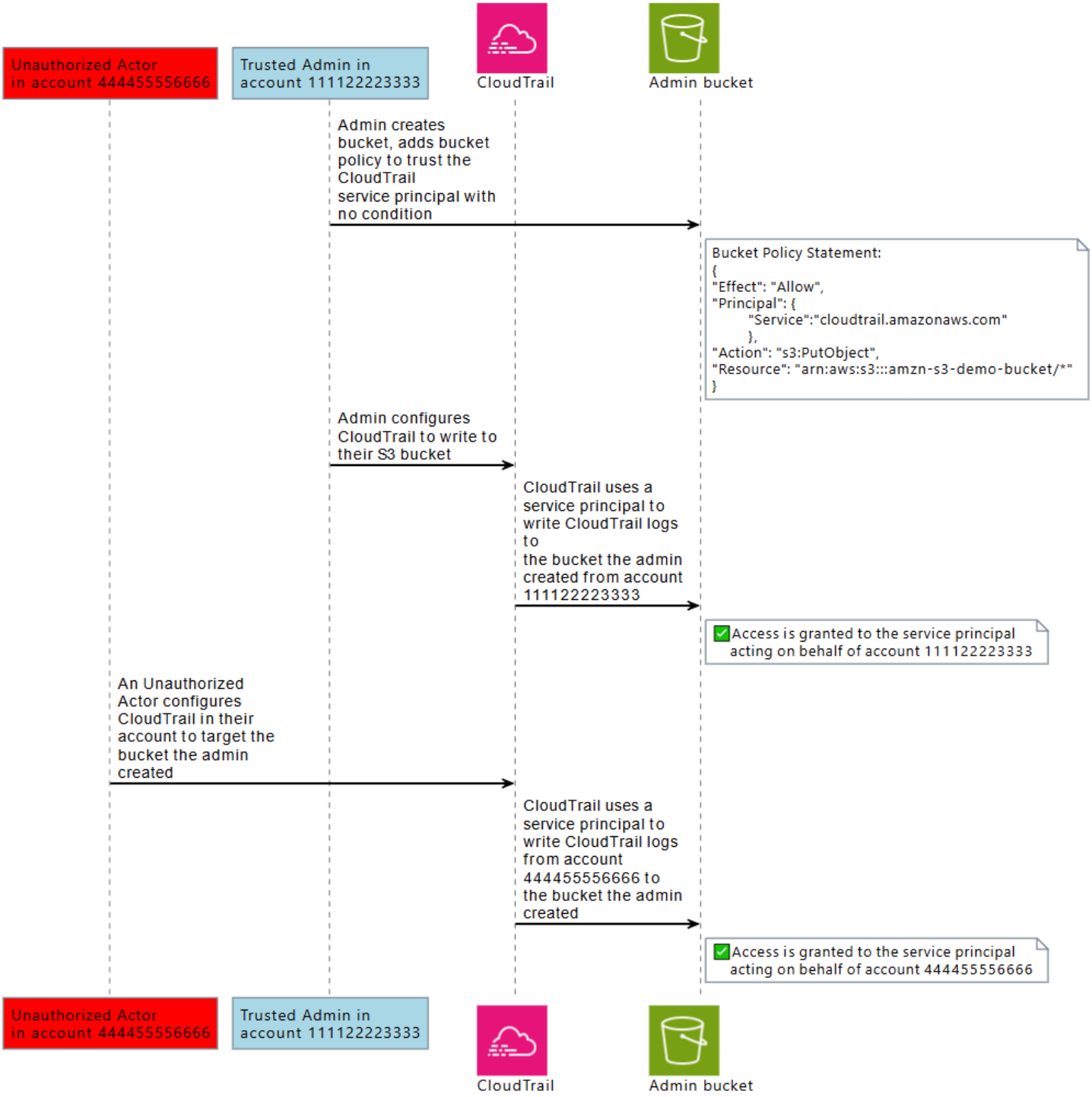


1. Seperti sebelumnya, ketika Anda mulai menggunakan layanan Example Corp, Anda memberikan ARN 1ExampleRole: ke AWS Example Corp.
2. Saat Example Corp menggunakan ARN peran itu untuk mengambil AWS peran 1 ExampleRole:, Example Corp menyertakan ID eksternal Anda (12345) dalam panggilan API. AssumeRole ID eksternal cocok dengan kebijakan kepercayaan peran, sehingga panggilan AssumeRole API berhasil dan Example Corp memperoleh kredensyal keamanan sementara untuk mengakses sumber daya di file Anda. Akun AWS
3. AWS Pelanggan lain juga mulai menggunakan layanan Example Corp, dan seperti sebelumnya, pelanggan ini juga menyediakan ARN 1ExampleRole: untuk AWS Example Corp untuk digunakan.
4. Tapi kali ini, ketika Example Corp mencoba untuk mengambil peran AWS 1: ExampleRole, ia memberikan ID eksternal yang terkait dengan pelanggan lain (67890). Pelanggan lain tidak bisa mengubah ini. Contoh Corp melakukan ini karena permintaan untuk menggunakan peran berasal dari pelanggan lain, jadi 67890 menunjukkan keadaan di mana Example Corp bertindak. Karena Anda menambahkan kondisi dengan ID eksternal Anda sendiri (12345) ke kebijakan kepercayaan AWS 1: ExampleRole, panggilan AssumeRole API gagal. Pelanggan lain tidak bisa mendapatkan akses tanpa izin ke sumber daya dalam akun Anda (ditunjukkan oleh “X” merah di diagram).

ID eksternal membantu mencegah pelanggan lain menipu Example Corp agar tanpa disadari mengakses sumber daya Anda.

Cross-service pencegahan wakil bingung

Diagram berikut menunjukkan masalah wakil kebingungan lintas layanan menggunakan contoh interaksi Amazon S3 CloudTrail dan Amazon, di mana aktor yang tidak sah CloudTrail menulis log ke ember Amazon S3 yang tidak diizinkan untuk mereka akses.

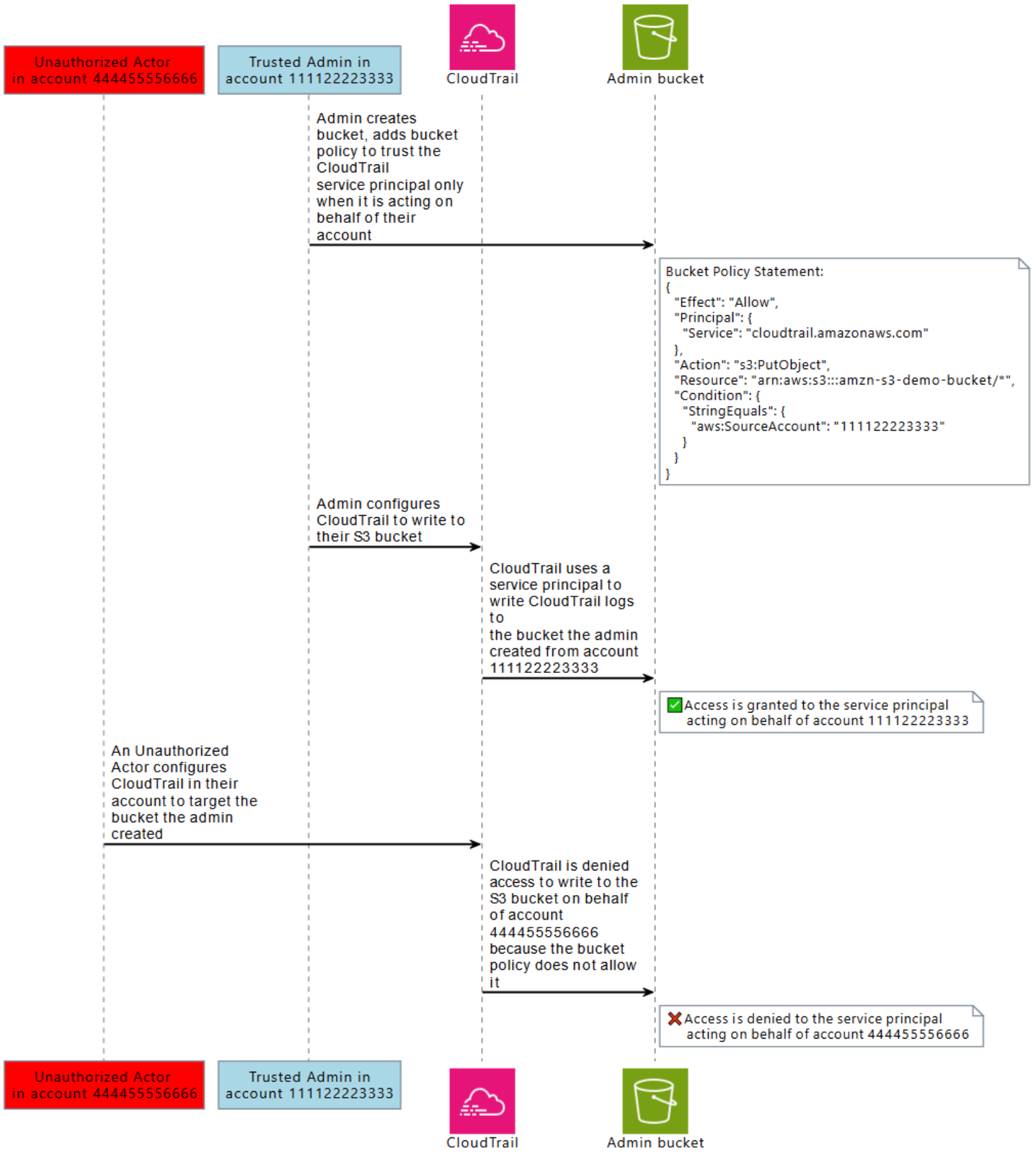


Untuk membantu melindungi dari aktor yang tidak berwenang dari menggunakan kepercayaan AWS kepala sekolah untuk mendapatkan akses ke sumber daya Anda, kepala AWS layanan menyertakan informasi tentang AWS sumber daya Akun AWS, dan AWS organisasi tempat mereka bertindak atas nama.

Informasi ini tersedia dalam nilai kunci kondisi global yang dapat digunakan dalam kebijakan sumber daya, atau kebijakan kontrol sumber daya untuk permintaan yang dibuat oleh prinsipal AWS layanan. Sebaiknya gunakanaw: SourceArn, aw: SourceAccountaws: SourceOrg ID, atau aw: SourceOrgPaths dalam kebijakan sumber daya Anda di mana pun kepala AWS layanan diberikan izin untuk mengakses salah satu sumber daya Anda. Kunci kondisi ini memungkinkan Anda untuk menguji kebijakan sumber daya Anda, atau kebijakan kontrol sumber daya yang dilakukan oleh prinsipal AWS layanan yang mengakses sumber daya Anda atas nama sumber AWS daya Akun AWS, atau yang Anda harapkan. AWS Organizations

- Gunakan `aws:SourceArn` untuk memungkinkan kepala AWS layanan mengakses sumber daya Anda atas nama sumber daya tertentu, seperti AWS CloudTrail jalur atau AppStream armada tertentu.
- Gunakan `aws:SourceAccount` untuk memungkinkan kepala AWS layanan mengakses sumber daya Anda atas nama tertentu Akun AWS.
- Gunakan `aws:SourceOrgID` untuk memungkinkan kepala AWS layanan mengakses sumber daya Anda atas nama tertentu AWS Organizations.
- Gunakan `aws:SourceOrgPaths` untuk memungkinkan kepala AWS layanan mengakses sumber daya Anda atas nama AWS Organizations jalur tertentu.

Diagram berikut menunjukkan skenario wakil kebingungan lintas layanan ketika sumber daya dikonfigurasi dengan kunci konteks kondisi `aws:SourceAccount` global, dan aktor yang tidak sah dari akun lain mencoba mengakses AWS sumber daya yang tidak dimaksudkan untuk mereka akses.



Menggunakan`aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, dan kunci kondisi `aws:SourceOrgPaths` global dalam kebijakan membantu Anda memastikan prinsip layanan

mengakses sumber daya Anda atas nama Anda. Sebaiknya gunakan kunci kondisi ini setiap kali akses ke salah satu sumber daya Anda diberikan kepada prinsipal AWS layanan.

> **Note**
>
> Beberapa Layanan AWS interaksi memiliki kontrol tambahan untuk membantu melindungi dari masalah wakil lintas layanan yang membingungkan yang menguji akses pengguna ke sumber daya. Misalnya, ketika hibah kunci KMS dikeluarkan untuk Layanan AWS, AWS KMS menggunakan konteks enkripsi yang terkait dengan sumber daya, dan hibah kunci untuk membantu melindungi dari masalah wakil lintas layanan yang membingungkan. Silakan lihat dokumentasi layanan yang Anda gunakan untuk informasi lebih lanjut tentang mekanisme khusus layanan yang dapat membantu menghindari risiko wakil lintas layanan yang membingungkan, dan apakah,`aws:SourceArn`, `aws:SourceAccountaws:SourceOrgID`, dan `aws:SourceOrgPaths` didukung.

Cross-service perlindungan wakil bingung dengan kebijakan berbasis sumber daya

Contoh kebijakan berikut memberikan `cloudtrail.amazonaws.com` akses utama layanan ke bucket Amazon S3, arn:aws:s3: ::amzn-s3-demo-bucket1, hanya jika kepala layanan bertindak atas nama 111122223333. Akun AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailAclCheck",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        },
        {
```

```
            "Sid": "AWSCloudTrailWrite",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/[optionalPrefix]/Logs/
myAccountID/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

Contoh kebijakan bucket ini memberikan `appstream.amazonaws.com` akses utama layanan ke skrip PowerShell examplefile.psh dalam s3://amzn-s3-demo-bucket2 hanya jika bertindak atas nama AppStream armada Amazon yang ditentukan, dengan menentukan armada arn dengan. `aws:SourceArn`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "appstream.amazonaws.com"
                ]
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/examplefile.psh",
            "Condition": {
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:appstream:us-
east-1:111122223333:fleet/ExampleFleetName"
                }
            }
        }
```

```
    ]
}
```

Cross-service perlindungan wakil bingung dengan kebijakan kontrol sumber daya

Anda dapat menggunakan kebijakan kontrol sumber daya (RCP) untuk menerapkan kontrol deputi lintas layanan yang membingungkan ke sumber daya yang didukung. Layanan AWS RCP memungkinkan Anda untuk secara terpusat menerapkan kontrol wakil yang membingungkan lintas layanan pada sumber daya Anda. Anda dapat menggunakan kunci kondisi seperti `aws:SourceOrgId` dan `aws:SourceOrgPaths` dengan RCP yang dilampirkan ke AWS Organizations, unit organisasi (OU) atau Akun AWS di organisasi Anda tanpa menambahkan pernyataan ke kebijakan berbasis sumber daya tertentu. Untuk informasi selengkapnya tentang RCP dan layanan yang didukung, lihat Kebijakan kontrol sumber daya (RCP) di Panduan Pengguna.AWS Organizations

Contoh berikut RCP menolak akses prinsipal AWS layanan ke bucket Amazon S3 di akun anggota Anda saat tidak sama dengan o-. `aws:SourceOrgID` ExampleOrg Izin yang sesuai harus ada dalam kebijakan berbasis sumber daya dari bucket S3 untuk mengizinkan Layanan AWS prinsipal yang sama dengan o-. `SourceOrgID` ExampleOrg

Kebijakan ini menerapkan kontrol hanya pada permintaan oleh service principals (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) yang memiliki `aws:SourceAccount` key present (`"Null": {"aws:SourceAccount": "false"}`), sehingga integrasi layanan yang tidak memerlukan penggunaan kunci kondisi dan panggilan oleh prinsipal Anda tidak terpengaruh. Jika kunci `aws:SourceAccount` kondisi hadir dalam konteks permintaan, kondisi Null akan mengevaluasi ke true, `aws:SourceOrgID` menyebabkan diberlakukan. Kami menggunakan `aws:SourceAccount` bukan `aws:SourceOrgID` di operator kondisi Null sehingga kontrol masih berlaku jika permintaan berasal dari akun yang bukan milik organisasi.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtectionForS3",
      "Effect": "Deny",
      "Principal": "*",
```

```
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-ExampleOrg"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Skenario umum untuk peran IAM

Seperti kebanyakan AWS fitur, Anda biasanya memiliki dua cara untuk menggunakan peran: interaktif di konsol IAM, atau secara terprogram dengan, Alat untuk Windows AWS CLI PowerShell, atau API.

- Pengguna IAM di akun Anda dengan menggunakan konsol IAM dapat beralih dari peran ke sementara menggunakan izin peran di konsol. Pengguna menyerahkan izin mereka dan mengambil izin yang ditetapkan untuk peran tersebut. Saat pengguna keluar dari peran, izin asli mereka dipulihkan.
- Aplikasi atau layanan yang ditawarkan oleh AWS (seperti Amazon EC2) dapat mengambil peran dengan meminta kredensil keamanan sementara untuk peran yang dapat digunakan untuk membuat permintaan terprogram. AWS Anda menggunakan peran dengan cara ini sehingga Anda tidak perlu berbagi atau mempertahankan kredensial keamanan jangka panjang (misalnya, dengan membuat pengguna IAM) untuk setiap entitas yang memerlukan akses ke sumber daya.

> **Note**
>
> Panduan ini menggunakan frasa beralih ke peran dan memegang peran secara bergantian.

Cara termudah untuk menggunakan peran adalah dengan memberikan izin kepada pengguna IAM Anda untuk beralih ke peran yang Anda buat dalam peran Anda sendiri atau yang lain. Akun AWS Mereka dapat dengan mudah beralih peran menggunakan konsol IAM untuk menggunakan izin yang biasanya tidak Anda inginkan, dan kemudian keluar dari peran tersebut untuk menyerahkan izin tersebut. Ini dapat membantu mencegah akses yang tidak disengaja atau modifikasi sumber daya sensitif.

Untuk penggunaan peran yang lebih kompleks, seperti memberikan akses ke aplikasi dan layanan, atau pengguna eksternal gabungan, Anda dapat menghubungi API `AssumeRole`. Panggilan API ini mengembalikan satu set kredensial sementara yang dapat digunakan aplikasi dalam panggilan API selanjutnya. Tindakan yang dicoba dengan kredensial sementara hanya memiliki izin yang diberikan oleh peran terkait. Sebuah aplikasi tidak harus "keluar" dari peran sebagaimana pengguna di konsol; melainkan aplikasi hanya berhenti menggunakan kredensial sementara dan melanjutkan melakukan panggilan dengan kredensial yang asli.

Pengguna gabungan masuk dengan menggunakan kredensil dari penyedia identitas (iDP). AWS kemudian memberikan kredensi sementara ke iDP tepercaya untuk diteruskan ke pengguna untuk disertakan dalam permintaan sumber daya berikutnya. AWS Kredensial tersebut memberikan izin yang diberikan kepada peran yang ditetapkan.

Bagian ini memberikan gambaran tentang skenario berikut:

- Menyediakan akses untuk pengguna IAM di salah satu Akun AWS yang Anda miliki untuk mengakses sumber daya di akun lain yang Anda miliki
- Menyediakan akses ke non beban AWS kerja
- Menyediakan akses ke pengguna IAM yang Akun AWS dimiliki oleh pihak ketiga
- Menyediakan akses untuk layanan yang ditawarkan oleh AWS ke AWS sumber daya
- Menyediakan akses untuk pengguna yang diautentikasi secara eksternal (federasi identitas)

Akses untuk pengguna IAM di yang lain Akun AWS yang Anda miliki

Anda dapat memberikan izin kepada pengguna IAM Anda untuk beralih ke peran dalam peran Anda Akun AWS atau ke peran yang ditentukan di peran lain Akun AWS yang Anda miliki.

> **Note**
>
> Jika Anda ingin memberikan akses ke akun yang tidak Anda miliki atau kontrol, lihat Akses ke Akun AWS dimiliki oleh pihak ketiga nanti dalam topik ini.

Bayangkan Anda memiliki instans Amazon EC2 yang sangat penting terhadap organisasi Anda. Alih-alih secara langsung memberikan izin kepada pengguna Anda untuk menghentikan instans, Anda dapat membuat peran atas hak istimewa tersebut. Kemudian, izinkan administrator untuk beralih ke peran saat mereka perlu menghentikan suatu instans. Melakukan hal ini akan menambah lapisan perlindungan berikut ke kejadian:

- Anda harus secara tegas memberikan izin kepada pengguna Anda untuk mengambil peran tersebut.
- Pengguna Anda harus secara aktif beralih ke peran menggunakan Konsol Manajemen AWS atau mengambil peran menggunakan AWS CLI atau AWS API.
- Anda dapat menambahkan perlindungan Multi-Factor Authentication (MFA) ke peran sehingga hanya pengguna yang masuk dengan perangkat MFA yang dapat memegang peran tersebut. Untuk mempelajari cara mengonfigurasi peran sehingga pengguna yang mengasumsikan peran tersebut harus diautentikasi terlebih dahulu menggunakan autentikasi Multi-Factor Authentication (MFA), lihat Akses API aman dengan MFA.

Kami menyarankan menggunakan pendekatan ini untuk menegakkan prinsip hak istimewa paling rendah. Ini berarti membatasi penggunaan izin yang ditingkatkan hanya pada saat diperlukan untuk tugas tertentu. Dengan peran, Anda dapat membantu mencegah perubahan yang tidak disengaja pada lingkungan sensitif, terutama jika Anda menggabungkannya dengan audit untuk membantu memastikan bahwa peran hanya digunakan saat diperlukan.

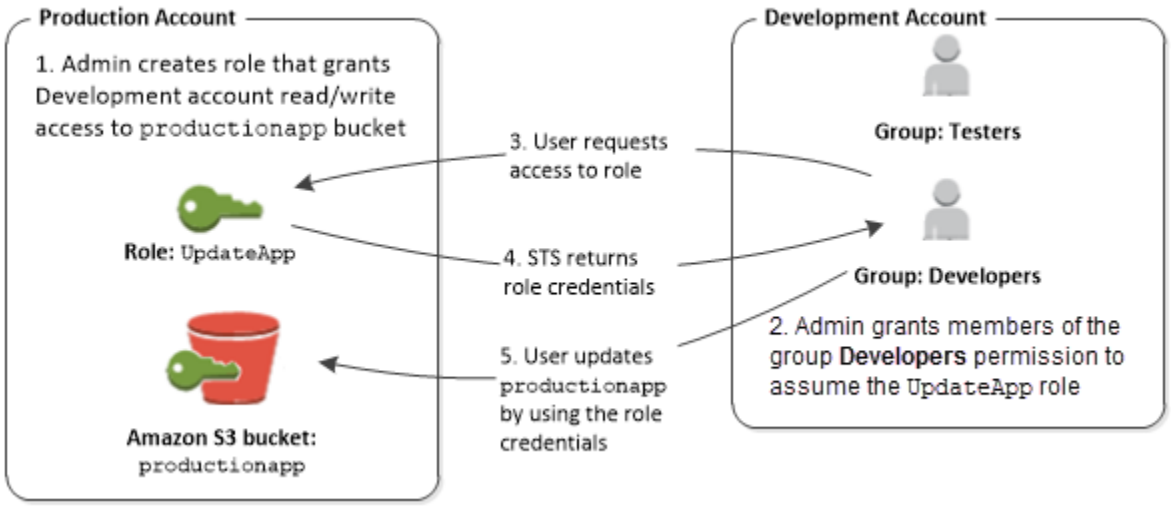
Saat Anda membuat peran untuk tujuan ini, Anda menetapkan akun berdasarkan ID yang penggunanya memerlukan akses dalam elemen `Principal` kebijakan kepercayaan peran. Kemudian, Anda dapat memberikan izin kepada pengguna tertentu dalam akun lain untuk beralih ke peran tersebut. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengasumsikan peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

Pengguna dalam satu akun dapat beralih ke peran dalam akun yang sama atau berbeda. Saat menggunakan peran tersebut, pengguna hanya dapat melakukan tindakan dan mengakses sumber

daya yang diizinkan dengan peran tersebut; izin pengguna yang asli ditangguhkan. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan.

Contoh skenario menggunakan akun pengembangan dan produksi terpisah

Bayangkan bahwa organisasi Anda memiliki banyak Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Pengguna di akun pengembangan mungkin kadang kala perlu mengakses sumber daya di akun produksi. Misalnya, Anda mungkin memerlukan akses akun silang saat mempromosikan pembaruan dari lingkungan pengembangan ke lingkungan produksi. Meskipun Anda dapat membuat identitas terpisah (dan kata sandi) untuk pengguna yang bekerja di kedua akun, mengelola kredensial untuk beberapa akun membuat manajemen identitas menjadi sulit. Dalam gambar berikut, semua pengguna dikelola dalam akun pengembangan, tetapi beberapa pengembang memerlukan akses terbatas ke akun produksi. Akun pengembangan memiliki dua grup: Penguji dan Developer, lalu setiap grup memiliki kebijakannya sendiri.



1. Di akun produksi, administrator menggunakan IAM untuk membuat peran `UpdateApp` dalam akun tersebut. Dalam peran tersebut, administrator mendefinisikan kebijakan kepercayaan yang menentukan akun pengembangan sebagai `Principal`, yang berarti bahwa pengguna sah dari akun pengembangan dapat menggunakan peran `UpdateApp`. Administrator juga menentukan kebijakan izin untuk peran yang menentukan izin baca dan tulis ke bucket Amazon S3 bernama`productionapp`.

   Kemudian, administrator membagikan informasi yang sesuai dengan siapa pun yang perlu menjalankan peran tersebut. Informasi tersebut adalah nomor akun dan nama peran (untuk pengguna AWS konsol) atau Nama Sumber Daya Amazon (ARN) (untuk AWS CLI atau akses AWS API). Peran ARN akan terlihat seperti `arn:aws:iam::123456789012:role/`

`UpdateApp`, di mana peran diberi nama `UpdateApp` dan peran tersebut dibuat pada nomor akun 123456789012.

> **Note**
>
> Administrator secara opsional dapat mengonfigurasi peran sehingga pengguna yang menerima peran tersebut harus terlebih dahulu diautentikasi menggunakan Multi-Factor Authentication (MFA) Untuk informasi selengkapnya, lihat Akses API aman dengan MFA.

2. Dalam akun pengembangan, administrator memberi anggota grup Pengembang izin untuk beralih ke peran. Ini dilakukan dengan memberikan izin grup Pengembang untuk memanggil AWS Security Token Service (AWS STS) `AssumeRole` API untuk `UpdateApp` peran tersebut. Pengguna IAM mana pun yang termasuk dalam grup Pengembang di akun pengembangan sekarang dapat beralih ke peran `UpdateApp` dalam akun produksi. Pengguna lain yang tidak berada dalam grup developer tidak memiliki izin untuk beralih ke peran tersebut sehingga tidak dapat mengakses bucket S3 dalam akun produksi.
3. Pengguna meminta untuk beralih ke peran:
   - AWS konsol: Pengguna memilih nama akun di bilah navigasi dan memilih Ber alih Peran. Pengguna menentukan ID akun (atau alias) dan nama peran. Atau, pengguna dapat memilih tautan yang dikirim melalui email oleh administrator. Tautan tersebut membawa pengguna ke halaman Beralih Peran dengan detail yang sudah diisi.
   - AWS API/AWS CLI: Pengguna dalam grup Pengembang dari akun pengembangan memanggil `AssumeRole` fungsi untuk mendapatkan kredenSIAL untuk `UpdateApp` peran tersebut. Pengguna menentukan ARN dari peran `UpdateApp` sebagai bagian dari panggilan. Jika pengguna di grup Testers melakukan permintaan yang sama, permintaan tersebut gagal karena Penguji tidak memiliki izin untuk menelepon `AssumeRole` untuk peran `UpdateApp` ARN.
4. AWS STS mengembalikan kredenSIAL sementara:
   - AWS console: AWS STS memverifikasi permintaan dengan kebijakan kepercayaan peran untuk memastikan bahwa permintaan berasal dari entitas tepercaya (yaitu: akun pengembangan). Setelah verifikasi, AWS STS mengembalikan kredenSIAL keamanan sementara ke AWS konsol.
   - API/CLI: AWS STS memverifikasi permintaan terhadap kebijakan kepercayaan peran untuk memastikan bahwa permintaan berasal dari entitas tepercaya (yaitu: akun Pengembangan). Setelah verifikasi, AWS STS mengembalikan kredentif keamanan sementara ke aplikasi.
5. KredenSIAL sementara memungkinkan akses ke AWS sumber daya:

- AWS konsol: AWS Konsol menggunakan kredenSIAL sementara atas nama pengguna untuk semua tindakan konsol berikutnya, dalam hal ini, untuk membaca dan menulis ke `productionapp` bucket. Konsol tidak dapat mengakses sumber daya lain di akun produksi. Saat pengguna keluar dari peran, izin pengguna kembali ke izin awal yang ada sebelum beralih ke peran.
- API/CLI: Aplikasi menggunakan kredenSIAL keamanan sementara untuk memperbarui `productionapp` bucket. Dengan kredensial keamanan sementara, aplikasi hanya dapat membaca dari dan menulis ke bucket `productionapp` dan tidak dapat mengakses sumber daya lain dalam akun Produksi. Aplikasi tidak perlu keluar dari peran, tetapi berhenti menggunakan kredensial sementara dan menggunakan kredensial asli pada panggilan API selanjutnya.

Sumber daya tambahan

Untuk informasi selengkapnya, lihat informasi berikut:

- Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM

Akses untuk non-beban AWS kerja

Peran IAM adalah objek di AWS Identity and Access Management (IAM) yang diberi izin. Ketika Anda mengasumsikan peran tersebut menggunakan identitas IAM atau identitas dari luar AWS, itu memberi Anda kredensi keamanan sementara untuk sesi peran Anda. Anda mungkin memiliki beban kerja yang berjalan di pusat data atau infrastruktur lain di luar AWS yang harus mengakses AWS sumber daya Anda. Alih-alih membuat, mendistribusikan, dan mengelola kunci akses jangka panjang, Anda dapat menggunakan AWS Identity and Access Management Peran Di Mana Saja (Peran IAM Di Mana Saja) untuk mengautentikasi non-beban kerja Anda.AWS IAM Roles Anywhere menggunakan sertifikat X.509 dari otoritas sertifikat (CA) Anda untuk mengautentikasi identitas dan menyediakan akses dengan aman Layanan AWS dengan kredensi sementara yang disediakan oleh peran IAM.

Untuk menggunakan Peran IAM Di Mana Saja

1. Siapkan CA menggunakan AWS Private Certificate Authorityatau menggunakan CA dari infrastruktur PKI Anda sendiri.

2. Setelah menyiapkan CA, Anda membuat objek di IAM Roles Anywhere yang disebut jangkar kepercayaan. Anchor ini membangun kepercayaan antara IAM Roles Anywhere dan CA Anda untuk otentikasi.
3. Anda kemudian dapat mengonfigurasi peran IAM yang ada, atau membuat peran baru yang mempercayai layanan IAM Roles Anywhere.
4. Otentikasi AWS beban non-kerja Anda dengan IAM Roles Anywhere menggunakan jangkar kepercayaan. AWS memberikan kredensil sementara AWS non-beban kerja ke peran IAM yang memiliki akses ke sumber daya Anda. AWS

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang menyediakan akses ke AWS non-beban kerja.

- Untuk informasi selengkapnya tentang mengonfigurasi Peran IAM Di Mana Saja, lihat Apa itu AWS Identity and Access Management Peran Di Mana Saja di Panduan Pengguna IAM Roles Anywhere.
- Untuk mempelajari cara menyiapkan infrastruktur kunci publik (PKI) untuk Peran IAM Di Mana Saja, lihat Peran IAM Di Mana Saja dengan otoritas sertifikat eksternal di Blog Keamanan.AWS

Akses ke Akun AWS dimiliki oleh pihak ketiga

Ketika pihak ketiga memerlukan akses ke AWS sumber daya organisasi Anda, Anda dapat menggunakan peran untuk mendelegasikan akses kepada mereka. Misalnya, pihak ketiga mungkin menyediakan layanan untuk mengelola sumber daya AWS Anda. Dengan peran IAM, Anda dapat memberikan akses kepada pihak ketiga ini ke AWS sumber daya Anda tanpa membagikan kredentif AWS keamanan Anda. Sebaliknya, pihak ketiga dapat mengakses AWS sumber daya Anda dengan mengambil peran yang Anda buat di sumber daya Anda Akun AWS. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengasumsikan peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

Pihak ketiga harus menyediakan Anda dengan informasi berikut untuk membuat peran yang dapat mereka tanggung:

- Akun AWS ID pihak ketiga. Anda menentukan Akun AWS ID mereka sebagai prinsipal ketika Anda menentukan kebijakan kepercayaan untuk peran tersebut.

- ID eksternal untuk dikaitkan secara unik dengan peran. ID eksternal dapat berupa pengenal apa pun yang hanya diketahui oleh Anda dan pihak ketiga. Misalnya, Anda dapat menggunakan ID faktur antara Anda dan pihak ketiga, tetapi tidak menggunakan sesuatu yang dapat ditebak, seperti nama atau nomor telepon pihak ketiga. Anda harus menentukan ID ini saat Anda menentukan kebijakan kepercayaan untuk peran tersebut. Pihak ketiga harus memberikan ID ini saat memegang peran tersebut.
- Izin yang dibutuhkan pihak ketiga untuk bekerja dengan AWS sumber daya Anda. Anda harus menentukan izin ini saat menentukan kebijakan izin peran. Kebijakan ini menetapkan tindakan apa yang dapat mereka lakukan dan sumber daya apa yang dapat mereka akses.

Setelah Anda membuat peran tersebut, Anda harus memberikan peran Amazon Resource Name (ARN) tersebut kepada pihak ketiga. Mereka memerlukan ARN peran Anda untuk menjalankan peran tersebut.

> **Important**
>
> Saat Anda memberikan akses kepada pihak ketiga ke AWS sumber daya Anda, mereka dapat mengakses sumber daya apa pun yang Anda tentukan dalam kebijakan. Penggunaan sumber daya Anda oleh mereka dibebankan kepada Anda. Pastikan bahwa Anda membatasi penggunaan sumber daya Anda dengan tepat.

ID eksternal untuk akses pihak ketiga

ID eksternal memungkinkan pengguna yang mengambil peran untuk menegaskan keadaan di mana mereka beroperasi. Itu ini juga memberikan cara bagi pemilik akun untuk mengizinkan peran tersebut untuk diasumsikan hanya dalam keadaan tertentu. Fungsi utama dari ID eksternal adalah untuk mengatasi dan mencegahMasalah confused deputy.

> **Important**
>
> AWS tidak memperlakukan ID eksternal sebagai rahasia. Setelah Anda membuat rahasia seperti pasangan kunci akses atau kata sandi AWS, Anda tidak dapat melihatnya lagi. ID eksternal untuk peran dapat dilihat oleh siapa pun yang memiliki izin untuk melihat peran tersebut.

Kapan saya harus menggunakan ID eksternal?

Gunakan ID eksternal dalam situasi-situasi berikut:

- Anda adalah Akun AWS pemilik dan Anda telah mengonfigurasi peran untuk pihak ketiga yang mengakses orang lain Akun AWS selain milik Anda. Anda harus meminta ID eksternal dari pihak ketiga yang disertakannya, ketika mereka mengasumsikan peran Anda. Kemudian Anda memeriksa ID eksternal tersebut dalam kebijakan kepercayaan peran Anda. Melakukannya memastikan bahwa pihak eksternal dapat mengasumsikan peran Anda hanya ketika bertindak atas nama Anda.
- Anda berada dalam posisi mengasumsikan peran atas nama pelanggan yang berbeda seperti Example Corp dalam skenario kami sebelumnya. Anda harus menetapkan ID eksternal unik untuk setiap pelanggan dan menginstruksikan mereka untuk menambahkan ID eksternal ke kebijakan kepercayaan peran mereka. Kemudian Anda harus memastikan bahwa Anda selalu menyertakan ID eksternal yang benar dalam permintaan Anda untuk mengasumsikan peran.

  Anda mungkin sudah memiliki pengidentifikasi unik untuk setiap pelanggan Anda, dan ID unik ini cukup untuk digunakan sebagai ID eksternal. ID eksternal bukan merupakan nilai khusus yang Anda perlukan untuk membuat secara eksplisit, atau melacak secara terpisah, hanya untuk tujuan ini.

  Anda harus selalu menentukan ID eksternal dalam `AssumeRole` panggilan API Anda. Selain saat pelanggan memberi Anda ARN peran, uji apakah Anda dapat mengasumsikan peran tersebut baik dengan atau tanpa ID eksternal yang benar. Jika Anda dapat mengasumsikan peran tersebut tanpa ID eksternal yang benar, jangan menyimpan ARN peran pelanggan dalam sistem Anda. Tunggu hingga pelanggan Anda memperbarui kebijakan kepercayaan peran untuk meminta ID eksternal yang benar. Dengan cara ini, Anda membantu pelanggan Anda untuk melakukan hal yang benar, yang membantu Anda semua melindungi diri dari masalah confused deputy.

Contoh skenario menggunakan ID eksternal

Misalnya, katakanlah Anda memutuskan untuk menyewa perusahaan pihak ketiga bernama Example Corp untuk memantau Anda Akun AWS dan membantu mengoptimalkan biaya. Untuk melacak pengeluaran harian Anda, Example Corp perlu mengakses AWS sumber daya Anda. Example Corp juga memantau banyak akun AWS lain untuk nasabah lain.

Jangan berikan akses Example Corp ke pengguna IAM dan kredenSIAL jangka panjangnya di AWS akun Anda. Sebaliknya, gunakan peran IAM dan kredensial keamanan sementaranya. Peran IAM

menyediakan mekanisme untuk memungkinkan pihak ketiga mengakses AWS sumber daya Anda tanpa perlu berbagi kredenSIAL jangka panjang (seperti kunci akses pengguna IAM).

Anda dapat menggunakan peran IAM untuk membangun hubungan tepercaya antara akun Anda Akun AWS dan Example Corp. Setelah hubungan ini terbentuk, anggota akun Example Corp dapat memanggil AWS Security Token Service AssumeRole API untuk mendapatkan kredenSIAL keamanan sementara. Anggota Example Corp kemudian dapat menggunakan kredenSIAL untuk mengakses AWS sumber daya di akun Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang AssumeRole dan operasi AWS API lainnya yang dapat Anda panggil untuk mendapatkan kredenSIAL keamanan sementara, lihatBandingkan AWS STS kredensialnya.

Berikut rincian skenario ini secara lebih detail:

1. Anda merekrut Example Corp, sehingga mereka membuat pengenal pelanggan unik untuk Anda. Mereka memberi Anda ID pelanggan unik ini dan Akun AWS nomor mereka. Anda memerlukan informasi ini untuk membuat peran IAM pada langkah berikutnya.

   > **Note**
   >
   > Example Corp dapat menggunakan nilai string apa pun yang mereka inginkan untuk ExternalId, asalkan unik untuk setiap pelanggan. Itu dapat berupa nomor akun pelanggan atau bahkan string karakter acak, selama tidak ada dua pelanggan yang memiliki nilai yang sama. Itu tidak dimaksudkan untuk menjadi 'rahasia'. Contoh Corp harus memberikan ExternalId nilai kepada setiap pelanggan. Yang penting adalah bahwa itu harus dibuat oleh Example Corp dan bukan pelanggan mereka untuk memastikan setiap ID eksternal unik.

2. Anda masuk AWS dan membuat peran IAM yang memberi Example Corp akses ke sumber daya Anda. Seperti peran IAM lainnya, peran ini memiliki dua kebijakan, kebijakan izin, dan kebijakan kepercayaan. Kebijakan kepercayaan peran menentukan siapa yang dapat mengasumsikan peran tersebut. Dalam skenario sampel kami, kebijakan menentukan Akun AWS jumlah Example Corp sebagai`Principal`. Hal ini memungkinkan identitas dari akun tersebut untuk mengambil peran. Selain itu, Anda menambahkan elemen `Condition` pada kebijakan kepercayaan. Ini `Condition`

menguji `ExternalId` kunci konteks untuk memastikan bahwa itu sesuai dengan ID pelanggan unik dari Example Corp. Misalnya:

```
   "Principal": {"AWS": "Example Corp's Akun AWS ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. Kebijakan izin untuk peran menentukan apa saja yang diperbolehkan oleh peran untuk dilakukan seseorang. Misalnya, Anda dapat menentukan bahwa peran tersebut dapat memperbolehkan seseorang untuk mengelola hanya sumber daya Amazon EC2 Amazon dan Amazon RDS Anda tetapi tidak dengan pengguna atau grup IAM Anda. Dalam skenario sampel kami, Anda menggunakan kebijakan izin untuk memberikan akses hanya-baca kepada Example Corp ke semua sumber daya dalam akun Anda.
4. Setelah Anda membuat peran tersebut, Anda menyediakan Amazon Resource Name (ARN) dari peran Anda kepada Example Corp.
5. Ketika Example Corp perlu mengakses AWS sumber daya Anda, seseorang dari perusahaan memanggil AWS `sts:AssumeRole` API. Panggilan mencakup ARN dari peran yang akan diasumsikan dan ExternalId parameter yang sesuai dengan ID pelanggan mereka.

Jika permintaan berasal dari seseorang yang menggunakan Example Corp Akun AWS, dan jika peran ARN dan ID eksternal benar, permintaan berhasil. Kemudian memberikan kredentif keamanan sementara yang dapat digunakan Example Corp untuk mengakses AWS sumber daya yang diizinkan peran Anda.

Dengan kata lain, ketika kebijakan peran mencakup ID eksternal, siapa pun yang ingin mengasumsikan peran tersebut haruslah prinsipal dalam peran tersebut dan harus menyertakan ID eksternal yang benar.

Poin kunci untuk ID eksternal

- Di lingkungan multi-penyewa di mana Anda mendukung banyak pelanggan dengan AWS akun yang berbeda, sebaiknya gunakan satu ID eksternal per. Akun AWS ID ini harus berupa string acak yang dihasilkan oleh pihak ketiga.
- Untuk mewajibkan pihak ketiga untuk memberikan ID eksternal saat mengasumsikan peran, perbarui kebijakan kepercayaan peran tersebut dengan ID eksternal pilihan Anda.

- Untuk memberikan ID eksternal saat Anda mengambil peran, gunakan AWS API AWS CLI or untuk mengambil peran itu. Untuk informasi lebih lanjut, lihat operasi API STS AssumeRole, atau operasi CLI STS assume-role.
- Nilai `ExternalId` harus memiliki minimal 2 karakter dan maksimal 1.224 karakter. Nilai harus berupa alfanumerik tanpa spasi. Alfanumerik dapat mencakup simbol berikut: plus (+), sama (=), koma (,), titik (.), di (@), titik dua (:), garis miring (/), dan tanda hubung (-).

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga.

- Untuk mempelajari cara mengizinkan orang lain melakukan tindakan di Anda Akun AWS, lihatMembuat peran menggunakan kebijakan kepercayaan khusus.
- Untuk mempelajari cara memberikan izin untuk beralih ke peran, lihat Berikan izin pengguna untuk beralih peran
- Untuk mempelajari cara membuat dan memberikan kredentif keamanan sementara kepada pengguna tepercaya, lihatIzin untuk kredensial keamanan sementara.

Akses ke AWS layanan

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk mengontrol apa yang dapat diakses oleh layanan tersebut. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut peran layanan. Ketika suatu peran melayani tujuan khusus untuk suatu layanan, itu dapat dikategorikan sebagai peran terkait layanan. Lihat dokumentasi AWS untuk setiap layanan guna melihat apakah layanan menggunakan peran dan untuk pelajari cara menetapkan peran untuk digunakan layanan.

Untuk detail tentang membuat peran untuk mendelegasikan akses ke layanan yang ditawarkan oleh AWS, lihatBuat peran untuk mendelegasikan izin ke AWS layanan.

Akses ke pengguna yang diautentikasi secara eksternal (federasi identitas)

Pengguna Anda mungkin sudah memiliki identitas di luar AWS, seperti di direktori perusahaan Anda. Jika pengguna tersebut perlu bekerja dengan AWS sumber daya (atau bekerja dengan aplikasi yang mengakses sumber daya tersebut), maka pengguna tersebut juga memerlukan kreden AWS tif keamanan. Anda dapat menggunakan peran IAM untuk menentukan izin bagi pengguna yang identitasnya difederasi dari organisasi Anda atau penyedia identitas pihak ketiga (IdP).

> **Note**
>
> Sebagai praktik terbaik keamanan, sebaiknya Anda mengelola akses pengguna di Pusat Identitas IAM dengan federasi identitas alih-alih membuat pengguna IAM. Untuk informasi tentang situasi tertentu di mana pengguna IAM diperlukan, lihat Kapan membuat pengguna IAM (bukan peran).

Set izin Pusat Identitas IAM menyediakan templat peran IAM yang dikelola Pusat Identitas IAM. Manajer akses akun — fitur IAM — memungkinkan Anda menetapkan peran IAM kustom Anda sendiri ke pengguna dan grup Pusat Identitas IAM, mengaktifkan kemampuan seperti kebijakan kepercayaan khusus, tag peran, dan jalur peran yang dapat dikonfigurasi. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.

Pengguna federasi dari aplikasi seluler atau berbasis web dengan Amazon Cognito

Jika Anda membuat aplikasi seluler atau berbasis web yang mengakses AWS sumber daya, aplikasi memerlukan kredentif keamanan untuk membuat permintaan terprogram. AWS Untuk sebagian besar skenario aplikasi seluler, kami menyarankan agar Anda menggunakan Amazon Cognito. Anda dapat menggunakan layanan ini dengan AWS Mobile SDK untuk iOS dan AWS Mobile SDK untuk Android dan Fire OS untuk membuat identitas unik bagi pengguna dan mengotentikasi mereka untuk akses aman ke AWS sumber daya Anda. Amazon Cognito mendukung penyedia identitas yang sama seperti yang tercantum di bagian berikutnya, dan juga mendukung identitas yang diautentikasi pengembang dan akses (tamu) yang tidak terotentikasi. Amazon Cognito juga menyediakan operasi API untuk mensinkronisasi data pengguna sehingga data terjaga saat pengguna berpindah antar perangkat. Untuk informasi selengkapnya, lihat Amazon Cognito untuk aplikasi seluler.

Menggabungkan pengguna dengan penyedia layanan identitas publik atau OpenID Connect

Bila memungkinkan, gunakan Amazon Cognito untuk skenario aplikasi berbasis web dan seluler. Amazon Cognito melakukan sebagian besar pekerjaan di balik layar dengan layanan penyedia identitas publik untuk Anda. Ia bekerja dengan layanan pihak ketiga yang sama dan juga mendukung masuk secara anonim. Namun, untuk skenario yang lebih lanjut, Anda dapat bekerja secara langsung dengan layanan pihak ketiga seperti Login with Amazon, Facebook, Google, atau IdP yang kompatibel dengan OpenID Connect (OIDC). Untuk informasi selengkapnya tentang menggunakan federasi OIDC menggunakan salah satu layanan ini, lihatFederasi OIDC.

Pengguna federasi dengan SAML 2.0

Jika organisasi Anda sudah menggunakan paket perangkat lunak penyedia identitas yang mendukung SAML 2.0 (Security Assertion Markup Language 2.0), Anda dapat membuat kepercayaan antara organisasi Anda sebagai penyedia identitas (IdP) dan AWS sebagai penyedia layanan. Anda kemudian dapat menggunakan SAML untuk menyediakan pengguna Anda dengan satu tanda masuk (SSO) federasi ke Konsol Manajemen AWS atau akses federasi untuk memanggil operasi API. AWS Misalnya, jika perusahaan Anda menggunakan Microsoft Active Directory dan Active Directory Federation Services, maka Anda dapat melakukan federasi menggunakan SAML 2.0. Untuk informasi selengkapnya tentang pengguna federasi dengan SAML 2.0, lihat Federasi SAML 2.0.

Pengguna federasi dengan membuat aplikasi broker identitas khusus

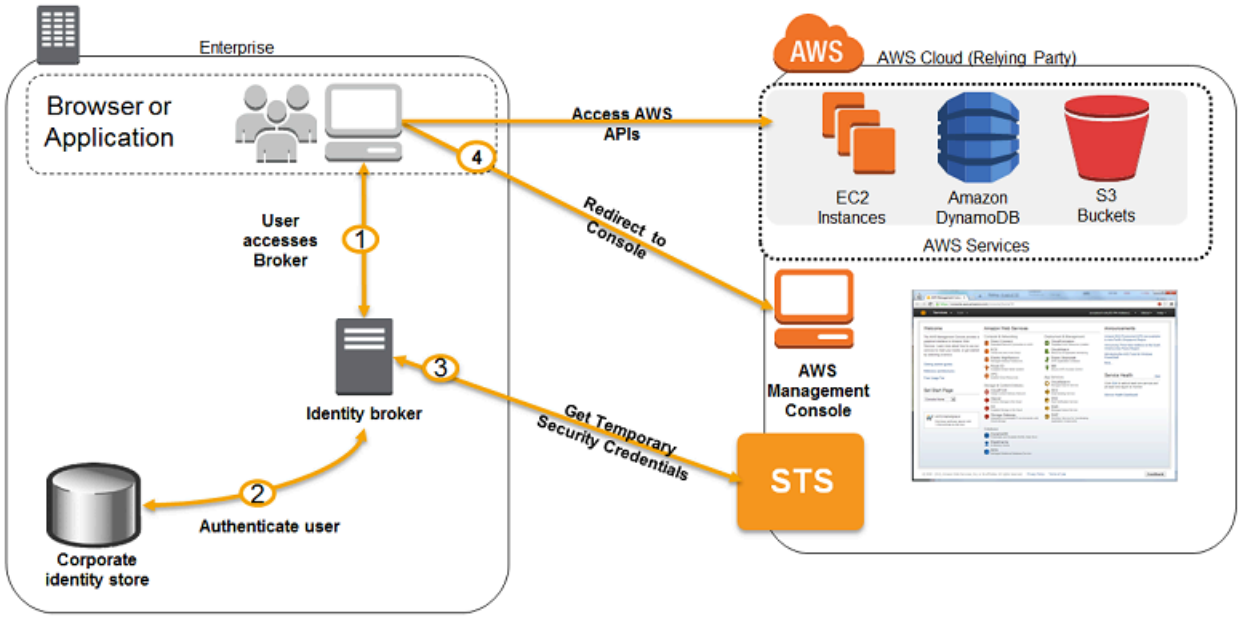
Jika penyimpanan identitas tidak kompatibel dengan SAML 2.0, maka Anda dapat membuat aplikasi broker identitas khusus untuk melakukan fungsi yang serupa. Aplikasi broker mengotentikasi pengguna, meminta kredenSIAL sementara untuk pengguna AWS, dan kemudian memberikannya kepada pengguna untuk mengakses AWS sumber daya.

Misalnya, Example Corp. memiliki banyak karyawan yang perlu menjalankan aplikasi internal yang mengakses AWS sumber daya perusahaan. Karyawan tersebut sudah memiliki identitas di sistem identitas dan autentikasi perusahaan, dan Example Corp. tidak ingin membuat pengguna IAM terpisah untuk setiap karyawan perusahaan.

Bob adalah pengembang di Example Corp. Untuk mengaktifkan aplikasi internal Example Corp untuk mengakses AWS sumber daya perusahaan, Bob mengembangkan aplikasi broker identitas khusus. Aplikasi memverifikasi bahwa karyawan dimasukkan ke dalam sistem identitas dan autentikasi Example Corp. yang ada, yang mungkin menggunakan LDAP, Active Directory, atau sistem lainnya. Aplikasi broker identitas kemudian mendapatkan kredensial keamanan sementara bagi karyawan. Skenario ini mirip dengan yang sebelumnya (aplikasi seluler yang menggunakan sistem otentikasi khusus), kecuali bahwa aplikasi yang membutuhkan akses ke AWS sumber daya semuanya berjalan dalam jaringan perusahaan, dan perusahaan memiliki sistem otentikasi yang ada.

Untuk mendapatkan kredensial keamanan sementara, aplikasi broker identitas menelepon `AssumeRole` atau `GetFederationToken` untuk memperoleh kredensial keamanan sementara, tergantung pada bagaimana Bob ingin mengelola kebijakan untuk pengguna dan kapan kredensial sementara harus kedaluwarsa. (Untuk informasi selengkapnya tentang perbedaan antara operasi API tersebut, lihat Kredensial keamanan sementara di IAM dan Izin untuk kredensial keamanan sementara.) Panggilan mengembalikan kredentif keamanan sementara yang terdiri dari ID kunci AWS akses, kunci akses rahasia, dan token sesi. Aplikasi broker identitas membuat kredensial

keamanan sementara ini tersedia bagi aplikasi internal perusahaan. Aplikasi kemudian dapat menggunakan kredensial sementara untuk membuat panggilan ke AWS secara langsung. Aplikasi menyimpan kredensial hingga kedaluwarsa, kemudian meminta set kredensial sementara yang baru. Gambar berikut mengilustrasikan skenario ini.



Skenario ini memiliki atribut berikut:

- Aplikasi pialang identitas memiliki izin untuk mengakses layanan token API (STS) IAM untuk membuat kredensial keamanan sementara.
- Aplikasi pialang identitas dapat memverifikasi bahwa karyawan diautentikasi dengan sistem autentikasi yang sudah ada.
- Pengguna bisa mendapatkan URL sementara yang memberi mereka akses ke AWS Management Console (yang disebut sebagai single sign-on).

Untuk informasi tentang membuat kredensial keamanan sementara, lihat Bandingkan AWS STS kredensialnya. Untuk informasi selengkapnya tentang prinsipal federasi SAML yang mendapatkan akses ke Konsol AWS Manajemen, lihat. Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS

Pembuatan peran IAM

Untuk membuat peran, Anda dapat menggunakan Konsol Manajemen AWS, AWS CLI, Alat untuk Windows PowerShell, atau IAM API.

Jika Anda menggunakan Konsol Manajemen AWS, wizard memandu Anda melalui langkah-langkah untuk membuat peran. Wizard memiliki langkah-langkah yang sedikit berbeda tergantung pada apakah Anda membuat peran untuk AWS layanan, untuk Akun AWS, atau untuk prinsipal federasi SAML atau OIDC.

Peran untuk pengguna IAM

Buat peran ini untuk mendelegasikan izin dalam Akun AWS atau ke peran yang ditentukan di peran lain Akun AWS yang Anda miliki. Pengguna dalam satu akun dapat beralih ke peran dalam akun yang sama atau berbeda. Saat menggunakan peran tersebut, pengguna hanya dapat melakukan tindakan dan mengakses sumber daya yang diizinkan dengan peran tersebut; izin pengguna yang asli ditangguhkan. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan.

Untuk informasi selengkapnya, lihat Buat peran untuk memberikan izin kepada pengguna IAM.

Untuk informasi selengkapnya tentang membuat peran untuk akses lintas akun, lihatMembuat peran menggunakan kebijakan kepercayaan khusus.

Peran untuk AWS layanan

Buat peran ini untuk mendelegasikan izin ke layanan yang dapat melakukan tindakan atas nama Anda. Peran layanan yang Anda berikan ke layanan harus memiliki kebijakan IAM dengan izin yang memungkinkan layanan untuk melakukan tindakan yang terkait dengan layanan tersebut. Izin yang berbeda diperlukan untuk setiap AWS layanan.

Untuk informasi selengkapnya tentang membuat peran layanan, lihatBuat peran untuk mendelegasikan izin ke AWS layanan.

Untuk informasi selengkapnya tentang membuat peran terkait layanan, lihat. Buat peran tertaut layanan

Peran untuk federasi identitas

Buat peran ini untuk mendelegasikan izin kepada pengguna yang sudah memiliki identitas di luar AWS. Saat Anda menggunakan penyedia identitas , Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Pengguna eksternal Anda masuk melalui IdP, dan Anda dapat memberikan izin identitas eksternal tersebut untuk menggunakan AWS sumber daya di akun Anda. Penyedia identitas membantu menjaga keamanan AWS akun Anda karena Anda tidak perlu mendistribusikan atau menyematkan kredentif keamanan jangka panjang, seperti kunci akses, di aplikasi Anda.

Lihat informasi yang lebih lengkap di Membuat peran untuk penyedia identitas pihak ketiga.

Buat peran untuk memberikan izin kepada pengguna IAM

Anda dapat menggunakan peran IAM untuk menyediakan akses ke AWS sumber daya Anda. Dengan peran IAM, Anda dapat membangun hubungan kepercayaan antara akun kepercayaan Anda dan akun AWS tepercaya lainnya. Akun trusting memiliki sumber daya yang akan diakses dan akun terpercaya berisi pengguna yang membutuhkan akses ke sumber daya. Namun, akun lain bisa saja memiliki sumber daya di akun Anda. Misalnya, akun kepercayaan mungkin mengizinkan akun tepercaya untuk membuat sumber daya baru, seperti membuat objek baru di bucket Amazon S3. Dalam hal ini, akun yang menciptakan sumber daya memiliki sumber daya dan mengendalikan siapa yang dapat mengakses sumber daya tersebut.

Setelah Anda membuat hubungan kepercayaan, pengguna IAM atau aplikasi dari akun tepercaya dapat menggunakan operasi `AssumeRole`API AWS Security Token Service (AWS STS). Operasi ini menyediakan kredensi keamanan sementara yang memungkinkan akses ke AWS sumber daya di akun Anda.

Akun-akun tersebut dapat dikendalikan oleh Anda, atau akun dengan pengguna-pengguna dapat dikendalikan oleh pihak ketiga. Jika akun lain dengan pengguna adalah Akun AWS yang tidak Anda kendalikan, maka Anda dapat menggunakan `externalId` atribut tersebut. ID eksternal dapat berupa kata atau nomor apa pun yang disepakati antara Anda dan administrator akun pihak ketiga. Opsi ini secara otomatis menambahkan persyaratan kepada kebijakan kepercayaan yang memperbolehkan pengguna untuk mengasumsikan peran hanya jika permintaan mencakup `sts:ExternalID` yang benar. Untuk informasi selengkapnya, lihat Akses ke Akun AWS dimiliki oleh pihak ketiga.

Untuk informasi tentang cara menggunakan peran untuk mendelegasikan izin, lihat Istilah dan konsep peran. Untuk informasi tentang menggunakan peran layanan untuk memperbolehkan layanan mengakses sumber daya di akun Anda, lihat Buat peran untuk mendelegasikan izin ke AWS layanan.

Membuat peran IAM (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk membuat peran yang dapat diasumsikan oleh pengguna IAM. Misalnya, asumsikan bahwa organisasi Anda memiliki banyak Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Untuk informasi tingkat tinggi tentang membuat peran yang memungkinkan pengguna di akun pengembangan mengakses sumber daya di akun produksi, lihatContoh skenario menggunakan akun pengembangan dan produksi terpisah.

Izin minimum

Untuk melakukan langkah-langkah berikut, Anda harus memiliki setidaknya izin IAM berikut:

- `access-analyzer:ValidatePolicy`
- `iam:AttachRolePolicy`
- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:GetAccountSummary`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:ListAccountAliases`
- `iam:ListAttachedRolePolicies`
- `iam:ListOpenIDConnectProviders`
- `iam:ListPolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSAMLProviders`

Console

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Pilih tipe Akun AWSperan.
4. Untuk membuat peran untuk akun Anda, pilih Akun ini. Untuk membuat peran untuk akun lain, pilih Lainnya Akun AWS dan masukkan ID Akun yang ingin Anda berikan akses ke sumber daya Anda.

   Administrator akun yang ditentukan dapat memberikan izin untuk mengasumsikan peran ini kepada setiap pengguna IAM dalam akun tersebut. Untuk melakukannya, administrator

melampirkan kebijakan kepada pengguna atau grup yang memberikan izin untuk tindakan `sts:AssumeRole`. Kebijakan tersebut harus menentukan ARN peran sebagai `Resource`.

5. Jika Anda memberikan izin kepada pengguna dari akun yang tidak Anda kendalikan, dan pengguna akan mengambil peran ini secara terprogram, pilih Memerlukan ID eksternal. ID eksternal dapat berupa kata atau nomor apa pun yang disepakati antara Anda dan administrator akun pihak ketiga. Opsi ini secara otomatis menambahkan persyaratan kepada kebijakan kepercayaan yang memperbolehkan pengguna untuk mengasumsikan peran hanya jika permintaan mencakup `sts:ExternalID` yang benar. Untuk informasi selengkapnya, lihat Akses ke Akun AWS dimiliki oleh pihak ketiga.

   > **Important**
   >
   > Memilih opsi ini membatasi akses ke peran hanya melalui AWS CLI, Alat untuk Windows PowerShell, atau AWS API. Ini karena Anda tidak dapat menggunakan AWS konsol untuk beralih ke peran yang memiliki `externalId` kondisi dalam kebijakan kepercayaannya. Namun, Anda dapat membuat jenis akses ini secara terprogram dengan menulis skrip atau aplikasi menggunakan SDK yang relevan. Untuk informasi lebih lanjut dan contoh skrip, lihat Cara Mengaktifkan Akses Lintas Akun ke Konsol Manajemen AWS dalam Blog Keamanan AWS .

6. Jika Anda ingin membatasi peran bagi pengguna yang masuk ke multi-factor authentication (MFA) (MFA), pilih Memerlukan MFA. Ini menambahkan persyaratan ke kebijakan kepercayaan peran yang memeriksa masuk MFA. Seorang pengguna yang ingin mengasumsikan peran tersebut harus masuk dengan kata sandi satu kali sementara dari perangkat MFA yang dikonfigurasi. Pengguna tanpa otentikasi MFA tidak dapat mengasumsikan peran tersebut. Untuk informasi lebih lanjut tentang MFA, lihat AWS Multi-factor otentikasi di IAM
7. Pilih Berikutnya.
8. IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin atau pilih Buat kebijakan untuk membuka tab peramban baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Pilih kotak centang di samping kebijakan izin yang Anda inginkan bagi siapa saja yang mengambil peran tersebut. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.

9. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

   Buka bagian Atur batasan izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.

10. Pilih Berikutnya.
11. Untuk Nama peran, masukkan nama peran Anda. Nama peran harus unik di dalam diri Anda Akun AWS. Bila nama peran digunakan dalam kebijakan atau sebagai bagian dari ARN, nama peran tersebut peka huruf besar/kecil. Saat nama peran muncul ke pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil. Karena berbagai entitas mungkin mereferensikan peran tersebut, Anda tidak dapat mengedit nama peran setelah peran tersebut dibuat.
12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
13. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan bagian izin untuk mengedit kasus penggunaan dan izin untuk peran tersebut. Anda akan dikembalikan ke halaman sebelumnya untuk melakukan pengeditan.
14. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
15. Tinjau peran dan kemudian pilih Buat peran.

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran dalam konsol, atau mengasumsikan peran tersebut secara terprogram. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

Membuat peran IAM ()AWS CLI

Membuat peran dari AWS CLI melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu AWS CLI Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran bagi akses lintas akun (AWS CLI)

1. Buat peran: aws iam create-role
2. Lampirkan kebijakan izin terkelola ke peran: aws iam attach-role-policy

   atau

   Buat kebijakan izin sebaris untuk peran tersebut: aws iam put-role-policy
3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran: aws iam put-role-permissions-boundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Contoh berikut menunjukkan dua langkah pertama, dan yang paling umum untuk membuat peran lintas akun dalam lingkungan sederhana. Contoh ini memungkinkan setiap pengguna dalam akun `123456789012` untuk mengasumsikan peran dan melihat `example_bucket` bucket Amazon S3. Contoh ini juga mengasumsikan bahwa Anda menggunakan komputer klien yang menjalankan Windows, dan telah mengonfigurasi antarmuka baris perintah Anda kredensial akun dan Wilayah Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi Antarmuka Baris AWS Perintah.

Dalam contoh ini, sertakan kebijakan kepercayaan berikut pada perintah pertama ketika Anda membuat peran tersebut. Kebijakan kepercayaan ini memperbolehkan pengguna dalam akun `123456789012` untuk mengasumsikan peran dengan menggunakan operasi `AssumeRole`, tetapi hanya jika pengguna memberikan otentikasi MFA dengan menggunakan parameter `SerialNumber` dan `TokenCode`. Untuk informasi lebih lanjut tentang MFA, lihat AWS Multi-factor otentikasi di IAM.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
```

```
            "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
        }
    ]
}
```

> **Important**
>
> Jika elemen `Principal` Anda memuat ARN untuk peran atau pengguna IAM tertentu, kemudian ARN diubah menjadi ID prinsipal unik saat kebijakan disimpan. Hal ini membantu memitigasi risiko seseorang meningkatkan izin mereka dengan menghapus dan membuat kembali peran atau pengguna. Anda biasanya tidak melihat ID ini di dalam konsol karena terdapat transformasi balik kembali ke ARN ketika kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus peran atau pengguna, maka ID utama muncul di konsol karena tidak AWS dapat lagi memetakannya kembali ke ARN. Oleh karena itu, jika Anda menghapus dan membuat ulang pengguna atau peran yang dirujuk di dalam elemen `Principal` kebijakan kepercayaan, Anda harus mengubah peran untuk mengganti ARN.

Saat menggunakan perintah kedua, Anda harus melampirkan kebijakan terkelola yang ada pada peran tersebut. Kebijakan izin berikut ini memperbolehkan siapa pun yang mengasumsikan peran untuk hanya melaksanakan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Untuk membuat peran `Test-UserAccess-Role` ini, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforacct123456789012.json`

ke folder `policies` di drive lokal `C:` Anda. Kemudian simpan kebijakan izin sebelumnya sebagai kebijakan terkelola pelanggan di nama `PolicyForRole` Anda Akun AWS . Kemudian Anda dapat menggunakan perintah berikut untuk membuat peran dan melampirkan kebijakan terkelola.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

Setelah Anda membuat peran dan memberinya izin untuk melakukan AWS tugas atau mengakses AWS sumber daya, setiap pengguna di `123456789012` akun dapat mengambil peran tersebut. Untuk informasi selengkapnya, lihat Beralih ke peran IAM (AWS CLI).

Membuat peran IAM (AWS API)

Membuat peran dari AWS API melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah yang dilakukan untuk Anda, tetapi dengan API Anda harus melakukan setiap langkah sendiri secara eksplisit. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran dalam kode (AWS API)

1. Buat peran: CreateRole

   Untuk kebijakan kepercayaan peran, Anda dapat menentukan lokasi file.
2. Lampirkan kebijakan izin terkelola pada peran: AttachRolePolicy

atau

Buat kebijakan izin inline untuk peran: PutRolePolicy

> **Important**
>
> Ingatlah bahwa ini hanya setengah bagian pertama dari konfigurasi yang diperlukan. Anda juga harus memberikan izin kepada pengguna individu dalam akun terpercaya untuk beralih ke peran. Untuk informasi selengkapnya tentang langkah ini, lihat Berikan izin pengguna untuk beralih peran.

3. (Opsional) Tambahkan atribut khusus ke peran dengan melampirkan tag: TagRole

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).

4. (Opsional) Tetapkan batas izin untuk peran tersebut: PutRolePermissionsBoundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Setelah Anda membuat peran dan memberinya izin untuk melakukan AWS tugas atau mengakses AWS sumber daya, Anda harus memberikan izin kepada pengguna di akun agar mereka dapat mengambil peran tersebut. Untuk informasi lebih lanjut tentang mengambil peran, lihat Beralih ke peran IAM (AWS API).

Membuat peran IAM ()AWS CloudFormation

Untuk informasi tentang membuat peran IAM AWS CloudFormation, lihat referensi sumber daya dan properti serta contoh di Panduan AWS CloudFormation Pengguna.

Untuk informasi selengkapnya tentang template IAM di AWS CloudFormation, lihat cuplikan AWS Identity and Access Management templat di Panduan Pengguna.AWS CloudFormation

Buat peran untuk mendelegasikan izin ke AWS layanan

Banyak AWS layanan mengharuskan Anda menggunakan peran untuk mengizinkan layanan mengakses sumber daya di layanan lain atas nama Anda. Peran yang diasumsikan layanan untuk melakukan tindakan atas nama Anda disebut peran layanan. Ketika peran melayani tujuan khusus untuk layanan, itu dikategorikan sebagai peran terkait layanan. Untuk melihat apa yang

didukung layanan dengan menggunakan peran yang terkait dengan layanan, atau apakah layanan mendukung segala bentuk kredensial sementara, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari bagaimana layanan individu menggunakan peran, pilih nama layanan dalam tabel untuk melihat dokumentasi untuk layanan tersebut.

Saat menyet `PassRole` el izin, Anda harus memastikan bahwa pengguna tidak melewati peran di mana peran tersebut memiliki lebih banyak izin daripada yang Anda inginkan untuk dimiliki pengguna. Misalnya, Alice mungkin tidak diizinkan untuk melakukan tindakan Amazon S3 apa pun. Jika Alice dapat meneruskan peran ke layanan yang memungkinkan tindakan Amazon S3, layanan dapat melakukan tindakan Amazon S3 atas nama Alice saat menjalankan pekerjaan.

Untuk informasi tentang cara peran membantu Anda untuk mendelegasikan izin, lihat Istilah dan konsep peran.

Izin peran layanan

Anda harus mengonfigurasikan izin untuk memperbolehkan entitas IAM (pengguna atau peran) untuk membuat atau mengubah peran terkait layanan.

> **Note**
>
> ARN untuk peran terkait layanan mencakup prinsipal layanan, yang ditunjukkan dalam kebijakan berikut sebagai. *`SERVICE-NAME`*`.amazonaws.com` Jangan mencoba menebak prinsip layanan, karena peka huruf besar/kecil dan formatnya dapat bervariasi di seluruh AWS layanan. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk mengizinkan entitas IAM membuat peran layanan tertentu

Tambahkan kebijakan berikut ke entitas IAM yang perlu membuat peran layanan tersebut. Kebijakan ini memungkinkan Anda untuk membuat peran layanan untuk layanan tertentu dan dengan nama yang spesifik. Anda dapat melampirkan kebijakan terkelola atau inline pada peran tersebut.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

Untuk mengizinkan entitas IAM membuat peran layanan apa pun

AWS menyarankan agar Anda hanya mengizinkan pengguna administratif untuk membuat peran layanan apa pun. Seseorang dengan izin untuk membuat peran dan melampirkan kebijakan apa pun dapat meningkatkan izin mereka sendiri. Sebagai gantinya, buat kebijakan yang memungkinkan mereka membuat hanya peran yang mereka butuhkan atau minta administrator membuat peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses keseluruhan Anda Akun AWS, gunakan kebijakan ter AdministratorAccess AWS kelola.

Untuk mengizinkan entitas IAM mengedit peran layanan

Tambahkan kebijakan berikut ke entitas IAM yang perlu mengubah peran layanan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
```

```
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk mengizinkan entitas IAM menghapus peran layanan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menghapus peran layanan tertentu.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Untuk mengizinkan entitas IAM menghapus peran layanan apa pun

AWS menyarankan agar Anda hanya mengizinkan pengguna administratif untuk menghapus peran layanan apa pun. Sebagai gantinya, buat kebijakan yang memungkinkan mereka menghapus hanya peran yang mereka butuhkan atau meminta administrator menghapus peran layanan atas nama mereka.

Untuk melampirkan kebijakan yang memungkinkan administrator mengakses keseluruhan Anda Akun AWS, gunakan kebijakan ter AdministratorAccess AWS kelola.

Menciptakan peran untuk sebuah AWS layanan (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk membuat peran untuk layanan. Karena beberapa layanan mendukung lebih dari satu peran layanan, lihat dokumentasi AWS agar layanan Anda dapat melihat kasus penggunaan mana yang dapat dipilih. Anda dapat mempelajari cara menetapkan kebijakan kepercayaan dan izin yang diperlukan untuk peran tersebut sehingga layanan dapat mengasumsikan peran tersebut atas nama Anda. Langkah-langkah yang dapat Anda gunakan untuk mengontrol izin untuk peran Anda dapat bervariasi, tergantung pada bagaimana layanan mendefinisikan kasus penggunaan, dan apakah Anda membuat peran terkait layanan atau tidak.

Console

Untuk membuat peran untuk Layanan AWS (Konsol IAM)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
3. Untuk Ti pe entitas tepercaya, pilih Layanan AWS.
4. Untuk Layanan atau kasus penggunaan, pilih layanan, lalu pilih kasus penggunaan. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang diperlukan layanan.
5. Pilih Berikutnya.
6. Untuk kebijakan Iz in, opsi tergantung pada kasus penggunaan yang Anda pilih:
   - Jika layanan menentukan izin untuk peran, Anda tidak dapat memilih kebijakan izin.
   - Pilih dari kumpulan kebijakan izin terbatas.
   - Pilih dari semua kebijakan izin.
   - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran tersebut.
7. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
   a. Buka bagian Tetapkan batas izin, lalu pilih Gunakan batas izin untuk mengontrol izin peran maksimum.

      IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda.

b. Pilih kebijakan yang akan digunakan untuk batas izin.

8. Pilih Berikutnya.
9. Untuk nama Per an, opsi tergantung pada layanan:

   - Jika layanan menentukan nama peran, Anda tidak dapat mengedit nama peran.
   - Jika layanan mendefinisikan awalan untuk nama peran, Anda dapat memasukkan akhiran opsional.
   - Jika layanan tidak menentukan nama peran, Anda dapat memberi nama peran tersebut.

   > **Important**
   >
   > Saat Anda memberi nama peran, perhatikan hal berikut:
   >
   > - Nama peran harus unik dalam nama Anda Akun AWS, dan tidak dapat dibuat unik berdasarkan kasus.
   >
   >   Misalnya, jangan membuat peran bernama keduanya **`PRODROLE`** dan**`prodrole`**. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dari ARN, nama peran tersebut peka huruf besar/kecil, namun ketika nama peran muncul kepada pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil.
   > - Anda tidak dapat mengedit nama peran setelah dibuat karena entitas lain mungkin mereferensikan peran tersebut.

10. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di bagian Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin, pilih Edit.
12. (Opsional) Untuk membantu mengidentifikasi, mengatur, atau mencari peran, tambahkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya di Panduan Pengguna IAM.
13. Tinjau peran lalu pilih Buat peran.

Membuat peran untuk layanan (AWS CLI)

Membuat peran dari AWS CLI melibatkan beberapa langkah. Ketika Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu Anda harus secara

eksplisit melakukan setiap langkah sendiri. AWS CLI Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Jika layanan yang Anda gunakan untuk bekerja adalah Amazon EC2, maka Anda juga harus membuat profil instans dan menambahkan peran kepadanya. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran untuk AWS layanan dari AWS CLI

1. Per `create-role` intah berikut membuat peran bernama Test-Role dan melampirkan kebijakan kepercayaan padanya:

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Lampirkan kebijakan izin terkelola ke peran: aws iam attach -role-policy.

   Misalnya, `attach-role-policy` perintah berikut melampirkan kebijakan terkel AWS ola yang dinam `ReadOnlyAccess` ai ke peran IAM bernama: `ReadOnlyRole`

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   ReadOnlyAccess --role-name ReadOnlyRole
   ```

   atau

   Buat kebijakan izin inline untuk peran: aws iam put-role-policy

   Untuk menambahkan kebijakan izin sebaris, lihat contoh berikut:

   ```
   aws iam put-role-policy --role-name Test-Role --policy-name
   ExamplePolicy --policy-document file://AdminPolicy.json
   ```

3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).

4. (Opsional) Tetapkan batas izin untuk peran tersebut: aws iam put-role-permissions-boundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan Amazon EC2, Anda harus menyimpan peran dalam profil instance. Profil instans

adalah wadah untuk peran yang dapat dilampirkan ke instans Amazon EC2 saat diluncurkan. Profil instans hanya dapat berisi satu , dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran menggunakan Konsol Manajemen AWS, profil instance dibuat untuk Anda dengan nama yang sama dengan peran tersebut. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil instance. Untuk informasi tentang cara meluncurkan instans EC2 dengan peran, lihat Mengontrol Ak ses ke Sumber Daya Amazon EC2 di Panduan Pengguna Amazon EC2.

Untuk membuat profil instance dan menyimpan peran di dalamnya (AWS CLI)

1. Buat profil instans: aws iam create-instance-profile
2. Tambahkan peran ke profil instans: aws iam add-role-to-instance-profile

AWS CLI Contoh perintah yang ditetapkan di bawah ini menunjukkan dua langkah pertama untuk membuat peran dan melampirkan izin. Itu juga menunjukkan dua langkah untuk membuat profil instans dan menambahkan peran ke profil. Contoh kebijakan kepercayaan ini memungkinkan layanan Amazon EC2 untuk mengambil peran dan melihat `example_bucket` bucket Amazon S3. Contoh ini juga mengasumsikan bahwa Anda sedang bekerja di komputer klien yang menjalankan Windows dan telah mengonfigurasi antarmuka baris perintah Anda dengan kredensial akun dan Wilayah Anda. Untuk informasi selengkapnya, lihat Meng onfigurasi Antar AWS muka Baris Perintah.

Dalam contoh ini, sertakan kebijakan kepercayaan berikut pada perintah pertama ketika Anda membuat peran tersebut. Kebijakan kepercayaan ini memungkinkan layanan Amazon EC2 untuk mengasumsikan peran.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Saat Anda menggunakan perintah kedua, Anda harus melampirkan kebijakan izin pada peran tersebut. Contoh kebijakan izin berikut memungkinkan peran untuk hanya mengasumsikan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Untuk membuat peran ini `Test-Role-for-EC2`, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforec2.json` dan kebijakan izin sebelumnya dengan nama `permissionspolicyforec2.json` ke `policies` direktori di lokasi Anda `C:`. Anda dapat menggunakan perintah berikut untuk membuat peran, melampirkan kebijakan, membuat profil instans, dan menambahkan peran ke profil instans.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Saat Anda meluncurkan instans EC2, tentukan nama profil instans di halaman Konfigurasi Detail Instans jika Anda menggunakan AWS konsol. Jika Anda menggunakan `aws ec2 run-instances` perintah CLI, tentukan `--iam-instance-profile` parameter.

Membuat peran untuk layanan (AWS API)

Membuat peran dari AWS API melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah yang dilakukan untuk Anda, tetapi dengan API Anda harus melakukan setiap langkah sendiri secara eksplisit. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Jika layanan yang Anda gunakan untuk bekerja adalah Amazon EC2, maka Anda juga harus membuat profil instans dan menambahkan peran kepadanya. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran untuk AWS layanan (AWS API)

1. Buat peran: CreateRole

   Untuk kebijakan kepercayaan peran, Anda dapat menentukan lokasi file.
2. Lampirkan kebijakan izin terkelola pada peran: AttachRolePolicy

   atau

   Buat kebijakan izin inline untuk peran: PutRolePolicy
3. (Opsional) Tambahkan atribut khusus ke peran dengan melampirkan tag: TagRole

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran tersebut: PutRolePermissionsBoundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Jika Anda akan menggunakan peran dengan Amazon EC2 atau AWS layanan lain yang menggunakan Amazon EC2, Anda harus menyimpan peran dalam profil instance. Profil contoh adalah wadah untuk peran. Profil instans hanya dapat berisi satu peran, dan batas tersebut tidak dapat ditingkatkan. Jika Anda membuat peran di Konsol Manajemen AWS, profil instance dibuat untuk Anda dengan nama yang sama dengan peran tersebut. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil instance. Untuk informasi tentang cara meluncurkan instans Amazon EC2 dengan peran, lihat Mengontrol Ak ses ke Sumber Daya Amazon EC2 di Panduan Pengguna Amazon EC2.

Untuk membuat profil instance dan menyimpan peran di dalamnya (AWS API)

1. Buat profil instans: CreateInstanceProfile
2. Tambahkan peran ke profil instans: AddRoleToInstanceProfile

Buat peran tertaut layanan

Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke layanan. AWS Service-linked peran telah ditentukan sebelumnya oleh layanan dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda. Layanan yang terhubung juga menentukan cara Anda membuat, memodifikasi, dan menghapus peran yang terkait dengan layanan. Layanan dapat secara otomatis membuat atau menghapus peran. Peran ini memungkinkan Anda membuat, memodifikasi, atau menghapus peran sebagai bagian dari wizard atau proses dalam layanan. Atau Anda harus menggunakan IAM untuk membuat atau menghapus peran tersebut. Apa pun metodenya, peran terkait layanan menyederhanakan proses pengaturan layanan karena Anda tidak perlu menambahkan izin secara manual untuk layanan untuk menyelesaikan tindakan atas nama Anda.

> **Note**
>
> Ingatlah bahwa peran layanan berbeda dari peran terkait layanan. Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM. Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat mengambil peran untuk melakukan tindakan atas nama Anda. Service-linked peran muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Layanan tertaut menentukan izin peran tertaut-layanannya, dan kecuali ditentukan lain, hanya layanan itu yang dapat mengambil peran tersebut. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menghapus sumber daya terkait. Ini membantu mencegah Anda menghapus izin untuk mengakses sumber daya secara tidak sengaja.

> **Tip**
>
> Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Service-linked izin peran

Anda harus mengonfigurasi izin untuk entitas IAM (pengguna atau peran) untuk mengizinkan pengguna atau peran membuat atau mengedit peran tertaut-layanan.

> **Note**
>
> ARN untuk peran terkait layanan mencakup prinsipal layanan, yang ditunjukkan dalam kebijakan di bawah ini sebagai *SERVICE-NAME*.amazonaws.com. Jangan mencoba menebak prinsip layanan, karena peka huruf besar/kecil dan formatnya dapat bervariasi di seluruh AWS layanan. Untuk melihat prinsipal layanan untuk suatu layanan, lihat dokumentasi peran yang terkait dengan layanan.

Untuk mengizinkan entitas IAM membuat peran terkait layanan tertentu

Tambahkan kebijakan berikut ke entitas IAM yang perlu membuat peran tertaut-layanan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
```

```
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Untuk mengizinkan entitas IAM membuat peran terkait layanan apa pun

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu membuat peran tertaut-layanan, atau peran layanan apa pun yang menyertakan kebijakan yang diperlukan. Pernyataan kebijakan ini tidak mengizinkan entitas IAM untuk melampirkan kebijakan ke peran tersebut.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Untuk mengizinkan entitas IAM mengedit deskripsi peran layanan apa pun

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu mengedit deskripsi peran tertaut-layanan, atau peran layanan apa pun.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Untuk mengizinkan entitas IAM menghapus peran terkait layanan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menghapus peran tertaut-layanan.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Untuk mengizinkan entitas IAM menghapus peran terkait layanan apa pun

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menghapus peran tertaut-layanan, tetapi bukan peran layanan.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Untuk mengizinkan entitas IAM meneruskan peran yang ada ke layanan

Beberapa AWS layanan memungkinkan Anda untuk meneruskan peran yang ada ke layanan, alih-alih membuat peran terkait layanan baru. Untuk melakukannya, pengguna harus memiliki izin untuk melewati peran tersebut dengan layanan tersebut. Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu meneruskan peran. Pernyataan kebijakan ini juga memungkinkan entitas untuk melihat daftar peran yang darinya mereka dapat memilih peran untuk diteruskan. Untuk informasi selengkapnya, lihat Memberikan izin pengguna untuk meneruskan peran ke AWS layanan.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
```

```
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Izin tidak langsung dengan peran terkait layanan

Izin yang diberikan oleh peran terkait layanan dapat ditransfer secara tidak langsung ke pengguna dan peran lain. Ketika peran terkait layanan digunakan oleh AWS layanan, peran terkait layanan tersebut dapat menggunakan izinnya sendiri untuk memanggil layanan lain. AWS Ini berarti bahwa pengguna dan peran dengan izin untuk memanggil layanan yang menggunakan peran terkait layanan mungkin memiliki akses tidak langsung ke layanan yang dapat diakses oleh peran terkait layanan tersebut.

Misalnya, saat Anda membuat instance Amazon RDS DB, peran yang ditautkan layanan untuk RDS secara otomatis dibuat jika peran tersebut belum ada. Peran terkait layanan ini memungkinkan RDS memanggil Amazon EC2, Amazon SNS, Amazon Logs, dan Amazon Kin CloudWatch esis atas nama Anda. Jika Anda mengizinkan pengguna dan peran di akun Anda untuk memodifikasi atau membuat database RDS, maka mereka mungkin dapat berinteraksi secara tidak langsung dengan Amazon EC2, Amazon SNS, CloudWatch log Amazon Logs, dan sumber daya Amazon Kinesis dengan memanggil RDS, karena RDS akan menggunakan peran terkait layanan untuk mengakses sumber daya tersebut.

Metode untuk membuat peran terkait layanan

Metode yang Anda gunakan untuk membuat peran terkait layanan tergantung pada layanannya. Dalam beberapa kasus, Anda tidak perlu membuat peran terkait layanan secara manual. Misalnya, ketika Anda menyelesaikan tindakan tertentu (seperti membuat sumber daya) dalam layanan, layanan tersebut mungkin membuat peran yang berkaitan dengan layanan bagi Anda. Atau jika Anda menggunakan layanan sebelum mulai mendukung peran tertaut-layanan, maka layanan tersebut mungkin secara otomatis membuat peran tersebut di akun Anda. Untuk mempelajari informasi selengkapnya, lihat Peran baru muncul di AWS akun.

Dalam kasus lain, layanan mungkin mendukung pembuatan peran yang ditautkan ke layanan secara manual menggunakan konsol layanan, API, atau CLI. Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan

cari layanan yang memiliki Ya di kolom Service-Linked Per an. Untuk mempelajari apakah layanan mendukung pembuatan peran tertaut-layanan, pilih tautan Ya untuk melihat dokumentasi peran tertaut-layanan untuk layanan itu.

Jika layanan tidak mendukung pembuatan peran, maka Anda dapat menggunakan IAM untuk membuat peran yang terhubung dengan layanan.

> **Important**
>
> Service-linked peran dihitung terhadap peran IAM Anda dalam Akun AWS batas tertentu, tetapi jika Anda telah mencapai batas, Anda masih dapat membuat peran terkait layanan di akun Anda. Hanya peran yang berkaitan dengan layanan yang dapat melebihi batas.

Membuat peran terkait layanan (konsol)

Sebelum Anda membuat peran yang ditautkan ke layanan di IAM, cari tahu apakah layanan yang ditautkan secara otomatis membuat peran yang ditautkan ke layanan. Selain itu, pelajari apakah Anda dapat membuat peran dari konsol layanan, API, atau CLI.

Untuk membuat peran terkait layanan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran. Kemudian, pilih Buat peran.
3. Pilih jenis peran AWS Layanan.
4. Pilih kasus penggunaan untuk layanan Anda. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang disyaratkan oleh layanan. Lalu, pilih Selanjutnya.
5. Pilih satu kebijakan izin atau lebih untuk dilampirkan ke peran tersebut. Bergantung pada kasus penggunaan yang Anda pilih, layanan mungkin melakukan salah satu hal berikut:
   - Tentukan izin yang digunakan oleh peran.
   - Memungkinkan Anda memilih dari seperangkat izin terbatas.
   - Memungkinkan Anda memilih dari izin apa pun.
   - Memungkinkan Anda memilih tidak ada kebijakan saat ini, membuat kebijakan nanti, lalu melampirkannya ke peran.

Pilih kotak centang di sebelah kebijakan yang menetapkan izin yang Anda inginkan untuk peran tersebut, lalu pilih Berikutnya.

> **Note**
>
> Izin yang Anda tentukan tersedia untuk setiap entitas yang menggunakan peran tersebut. Secara default, peran tidak memiliki izin.

6. Untuk Nama peran, tingkat penyesuaian nama peran ditentukan oleh layanan. Jika layanan mendefinisikan nama peran, maka opsi ini tidak dapat diedit. Dalam kasus lain, layanan mungkin menentukan awalan untuk peran dan memungkinkan Anda memasukkan akhiran opsional.

   Jika memungkinkan, masukkan akhiran nama peran untuk ditambahkan ke nama default. Akhiran ini membantu Anda mengidentifikasi tujuan peran ini. Nama peran harus unik di akun AWS Anda. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`<service-linked-role-name>_SAMPLE`** dan **`<service-linked-role-name>_sample`**. Anda tidak dapat mengubah nama peran setelah dibuat karena berbagai entitas mungkin mereferensikan peran tersebut.
7. (Opsional) Untuk Deskripsi, edit deskripsi untuk peran terkait layanan baru.
8. Anda tidak dapat melampirkan tag ke peran terkait layanan selama pembuatan. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
9. Tinjau peran, lalu pilih Buat peran.

Membuat peran terkait layanan (AWS CLI)

Sebelum membuat peran terkait layanan di IAM, cari tahu apakah layanan tertaut secara otomatis membuat peran terkait layanan dan apakah Anda dapat membuat peran dari CLI layanan. Jika CLI layanan tidak didukung, Anda dapat menggunakan perintah IAM untuk membuat peran terkait layanan dengan kebijakan kepercayaan dan kebijakan sebaris yang diperlukan layanan untuk menjalankan peran tersebut.

Untuk membuat peran terkait layanan ()AWS CLI

Jalankan perintah berikut:

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Membuat peran terkait layanan (AWS API)

Sebelum membuat peran terkait layanan di IAM, cari tahu apakah layanan tertaut secara otomatis membuat peran terkait layanan dan apakah Anda dapat membuat peran dari API layanan. Jika API layanan tidak didukung, Anda dapat menggunakan AWS API untuk membuat peran terkait layanan dengan kebijakan kepercayaan dan kebijakan sebaris yang diperlukan layanan untuk mengambil peran tersebut.

Untuk membuat peran terkait layanan (API)AWS

Gunakan panggilan API CreateServiceLinkedRole. Dalam permintaan, sebutkan nama layanan *SERVICE_NAME_URL*`.amazonaws.com`.

Misalnya, untuk membuat peran terkait layanan Lex Bots, gunakan `lex.amazonaws.com`.

Membuat peran untuk penyedia identitas pihak ketiga

Anda dapat menggunakan penyedia identitas alih-alih membuat pengguna IAM di situs Anda Akun AWS. Dengan penyedia identitas (iDP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan penyedia identitas, lihat Penyedia identitas dan federasi ke AWS.

Membuat peran untuk prinsipal federasi OIDC dan SAMP (konsol)

Prosedur untuk membuat peran tergantung pada pilihan penyedia pihak ketiga Anda:

- Untuk OpenID Connect (OIDC), lihat. Membuat peran untuk federasi OpenID Connect (konsol)
- Untuk SAML 2.0, lihat Buat peran untuk federasi SAML 2.0 (konsol).

Membuat peran untuk akses gabungan (AWS CLI)

Langkah-langkah untuk membuat peran untuk penyedia identitas yang didukung (OIDC atau SAML) dari AWS CLI adalah identik. Perbedaannya ada dalam konten kebijakan kepercayaan yang Anda buat dalam langkah-langkah prasyarat. Mulai dengan mengikuti langkah-langkah dalam bagian Prasyarat untuk jenis penyedia yang Anda gunakan:

- Untuk penyedia OIDC, lihat Prasyarat untuk membuat peran untuk OIDC.
- Untuk penyedia SAML, lihat Prasyarat untuk membuat peran untuk SAML.

Membuat peran dari AWS CLI melibatkan beberapa langkah. Saat Anda menggunakan konsol untuk membuat peran, banyak langkah dilakukan untuk Anda, tetapi dengan itu AWS CLI Anda harus secara eksplisit melakukan setiap langkah sendiri. Anda harus membuat peran dan kemudian menetapkan kebijakan izin untuk peran tersebut. Atau, Anda juga dapat mengatur batas izin untuk peran Anda.

Untuk membuat peran (AWS CLI)

1. Buat peran: aws iam create-role
2. Lampirkan kebijakan izin ke peran: aws iam attach-role-policy

   atau

   Buat kebijakan izin sebaris untuk peran tersebut: aws iam put-role-policy
3. (Opsional) Tambahkan atribut khusus ke peran tersebut dengan melampirkan tag: aws iam tag-role

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran: aws iam put-role-permissions-boundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Contoh berikut menunjukkan dua langkah pertama, dan paling umum, untuk membuat peran penyedia identitas dalam lingkungan yang sederhana. Contoh ini memungkinkan setiap pengguna dalam akun `123456789012` untuk mengasumsikan peran dan melihat `example_bucket` bucket Amazon S3. Contoh ini juga mengasumsikan bahwa Anda menjalankan AWS CLI pada komputer yang menjalankan Windows, dan telah mengkonfigurasi AWS CLI dengan kredensi Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS Command Line Interface.

Contoh kebijakan kepercayaan berikut ini dirancang untuk aplikasi seluler jika pengguna masuk dengan menggunakan Amazon Cognito. Dalam contoh ini, *`us-east:12345678-ffff-ffff-ffff-123456`* mewakili ID kumpulan identitas yang ditetapkan oleh Amazon Cognito.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

Kebijakan izin berikut ini memperbolehkan siapa pun yang mengasumsikan peran untuk hanya melaksanakan tindakan `ListBucket` pada `example_bucket` bucket Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Untuk membuat peran ini `Test-Cognito-Role`, Anda harus terlebih dahulu menyimpan kebijakan kepercayaan sebelumnya dengan nama `trustpolicyforcognitofederation.json` dan kebijakan izin sebelumnya dengan nama `permspolicyforcognitofederation.json` ke `policies` direktori di lokasi Anda `C:`. Kemudian Anda dapat menggunakan perintah berikut untuk membuat peran dan melampirkan kebijakan inline.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json
```

```
# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Membuat peran untuk akses federasi (AWS API)

Langkah-langkah untuk membuat peran untuk penyedia identitas yang didukung (OIDC atau SAML) dari AWS CLI adalah identik. Perbedaannya ada dalam konten kebijakan kepercayaan yang Anda buat dalam langkah-langkah prasyarat. Mulai dengan mengikuti langkah-langkah dalam bagian Prasyarat untuk jenis penyedia yang Anda gunakan:

- Untuk penyedia OIDC, lihat Prasyarat untuk membuat peran untuk OIDC.
- Untuk penyedia SAML, lihat Prasyarat untuk membuat peran untuk SAML.

Untuk membuat peran (AWS API)

1. Buat peran: CreateRole
2. Lampirkan kebijakan izin pada peran: AttachRolePolicy

   atau

   Buat kebijakan izin inline untuk peran: PutRolePolicy
3. (Opsional) Tambahkan atribut khusus ke peran dengan melampirkan tag: TagRole

   Untuk informasi selengkapnya, lihat Mengelola tag pada peran IAM (AWS CLI atau AWS API).
4. (Opsional) Tetapkan batas izin untuk peran tersebut: PutRolePermissionsBoundary

   Batas izin mengontrol izin maksimum yang dapat dimiliki sebuah peran. Batas izin adalah AWS fitur lanjutan.

Membuat peran untuk federasi OpenID Connect (konsol)

Anda dapat menggunakan penyedia identitas federasi OpenID Connect (OIDC) alih-alih membuat AWS Identity and Access Management pengguna di Anda. Akun AWS Dengan penyedia identitas (IdP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan IdPs, lihatPenyedia identitas dan federasi ke AWS.

Prasyarat untuk membuat peran untuk OIDC

Sebelum Anda dapat membuat peran untuk federasi OIDC, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan untuk membuat peran untuk federasi OIDC

1. Daftar dengan satu atau lebih layanan yang menawarkan identitas OIDC federasi. Jika Anda membuat aplikasi yang memerlukan akses ke AWS sumber daya, Anda juga mengonfigurasi aplikasi Anda dengan informasi penyedia. Jika Anda melakukannya, penyedia memberi Anda ID aplikasi atau audiens yang unik untuk aplikasi Anda. (Penyedia berbeda menggunakan terminologi berbeda untuk proses ini. Panduan ini menggunakan istilah konfigurasi untuk proses mengidentifikasi aplikasi Anda dengan penyedia.) Anda dapat mengonfigurasi beberapa aplikasi dengan setiap penyedia, atau beberapa penyedia dengan satu aplikasi. Lihat informasi tentang penggunaan penyedia identitas sebagai berikut:

   - Login dengan Amazon Developer Center
   - Tambahkan Login Facebook ke Aplikasi atau Website Anda di situs pengembang Facebook.
   - Menggunakan OAuth 2.0 untuk Login (OpenID Connect) di situs developer Google.
2. Setelah Anda menerima informasi yang diperlukan dari IdP, buat IdP di IAM. Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di IAM.

   > **Important**
   >
   > Jika Anda menggunakan IDP OIDC dari Google, Facebook, atau Amazon Cognito, jangan membuat IAM IDP terpisah di. Konsol Manajemen AWS Penyedia identitas OIDC ini sudah dibangun AWS dan tersedia untuk Anda gunakan. Lewati langkah ini dan buat peran baru menggunakan IdP Anda di langkah berikut.

3. Siapkan kebijakan untuk peran yang akan diasumsikan IdP-authenticated pengguna. Sebagaimana peran apa pun, peran untuk aplikasi seluler mencakup dua kebijakan. Salah satunya adalah kebijakan kepercayaan yang menentukan siapa yang dapat mengasumsikan peran tersebut. Lainnya adalah kebijakan izin yang menentukan tindakan AWS dan sumber daya yang diperbolehkan atau ditolak untuk diakses aplikasi seluler.

   Untuk web IdPs, kami sarankan Anda menggunakan Amazon Cognito untuk mengelola identitas. Dalam hal ini, gunakan kebijakan kepercayaan yang serupa dengan contoh ini.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

Ganti `us-east-2:12345678-abcd-abcd-abcd-123456` dengan ID kumpulan identitas yang diberikan Amazon Cognito kepada Anda.

Jika Anda mengonfigurasi IDP OIDC secara manual, saat membuat kebijakan kepercayaan, Anda harus menggunakan tiga nilai yang memastikan bahwa hanya aplikasi Anda yang dapat mengambil peran tersebut:

- Untuk elemen `Action`, gunakan tindakan `sts:AssumeRoleWithWebIdentity`.
- Untuk elemen `Principal`, gunakan string `{"Federated":`*`providerUrl/providerArn`*`}`.
  - Untuk beberapa OIDC umum IdPs, ini *`providerUrl`* adalah URL. Contoh-contoh berikut mencakup metode untuk menentukan prinsip untuk beberapa umum IdPs:

    `"Principal":{"Federated":"cognito-identity.amazonaws.com"}`

    `"Principal":{"Federated":"www.amazon.com"}`

    `"Principal":{"Federated":"graph.facebook.com"}`

    `"Principal":{"Federated":"accounts.google.com"}`
  - Untuk penyedia OIDC lainnya, gunakan Amazon Resource Name (ARN) dari IDP OIDC yang Anda buatStep 2, seperti contoh berikut:

```
"Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
server.example.com"}
```

- Untuk elemen `Condition` , gunakan syarat `StringEquals` untuk membatasi izin. Uji ID kumpulan identitas untuk Amazon Cognito) atau ID aplikasi untuk penyedia lain. ID kumpulan identitas harus sesuai dengan ID aplikasi yang Anda terima saat mengonfigurasi aplikasi dengan IdP. Kecocokan antar ID ini memastikan bahwa permintaan berasal dari aplikasi Anda.

> **Note**
>
> Peran IAM untuk kumpulan identitas Amazon Cognito mempercayai prinsipal layanan `cognito-identity.amazonaws.com` untuk mengambil peran tersebut. Peran jenis ini harus berisi setidaknya satu kunci kondisi untuk membatasi prinsipal yang dapat mengambil peran tersebut.
> Pertimbangan tambahan berlaku untuk kumpulan identitas Amazon Cognito yang mengambil peran I AM lintas akun. Kebijakan kepercayaan dari peran ini harus menerima prinsip `cognito-identity.amazonaws.com` layanan dan harus berisi kunci `aud` kondisi untuk membatasi asumsi peran kepada pengguna dari kumpulan identitas yang Anda inginkan. Kebijakan yang mempercayai kumpulan identitas Amazon Cognito tanpa kondisi ini menimbulkan risiko bahwa pengguna dari kumpulan identitas yang tidak diinginkan dapat mengambil peran tersebut. Untuk informasi selengkapnya, lihat Kebijakan kepercayaan untuk peran IAM dalam otentikasi Dasar (Klasik) di Panduan Pengembang Amazon Cognito.

Buat elemen kondisi yang mirip dengan salah satu contoh berikut, tergantung pada IdP yang Anda gunakan:

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Untuk penyedia OIDC, gunakan URL IdP OIDC yang memenuhi syarat sepenuhnya dengan kunci konteks `aud`, seperti contoh berikut:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Nilai-nilai untuk prinsipal dalam kebijakan kepercayaan untuk peran tersebut khusus untuk IdP. Peran untuk OIDC hanya dapat menentukan satu prinsipal. Oleh karena itu, jika aplikasi seluler memungkinkan pengguna masuk dari lebih dari satu IdP, buat peran terpisah untuk setiap IdP yang ingin Anda dukung. Buat kebijakan kepercayaan terpisah untuk setiap IdP.

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Login dengan Amazon, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh, *amzn1.application-oa2-123456* mewakili ID aplikasi yang ditetapkan Amazon saat Anda mengonfigurasi aplikasi menggunakan Login dengan Amazon.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Facebook, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *111222333444555* mewakili ID aplikasi yang ditetapkan Facebook.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Google, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *666777888999000* mewakili ID aplikasi yang ditetapkan Google.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Jika pengguna menggunakan aplikasi seluler untuk masuk dari Amazon Cognito, contoh kebijakan kepercayaan berikut akan berlaku. Dalam contoh ini, *us-east:12345678-ffff-ffff-ffff-123456* mewakili ID kumpulan identitas yang ditetapkan Amazon Cognito.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-
identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Membuat peran untuk OIDC

Setelah Anda menyelesaikan prasyarat, Anda dapat membuat peran dalam IAM. Untuk penyedia identitas OpenID Connect (OIDC) bersama yang diakui (IdPs), IAM memerlukan evaluasi eksplisit klaim spesifik dalam Token Web JSON (JWT) yang dikenal sebagai kontrol penyedia identitas. Untuk informasi selengkapnya tentang OIDC mana yang IdPs memiliki kontrol penyedia identitas, lihat. Identity-provider kontrol untuk penyedia OIDC bersama

Prosedur berikut menjelaskan cara membuat peran untuk federasi OIDC di Konsol Manajemen AWS. Untuk membuat peran dari AWS API AWS CLI or, lihat prosedur diMembuat peran untuk penyedia identitas pihak ketiga.

> **Important**
>
> Jika Anda menggunakan Amazon Cognito, gunakan konsol Amazon Cognito untuk mengatur peran. Jika tidak, gunakan konsol IAM untuk membuat peran federasi OIDC.

Untuk membuat peran IAM untuk federasi OIDC

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Peran lalu pilih Buat peran.
3. Pilih identitas Web sebagai jenis entitas tepercaya dan pilih Ber ikutnya.
4. Untuk penyedia Identitas, pilih IdP untuk peran Anda:

   - Jika Anda ingin membuat peran untuk IdP web individual, pilih Login dengan Amazon, Facebook, atau Google.

     > **Note**
     >
     > Anda harus membuat peran terpisah untuk setiap IdP yang ingin Anda dukung.

   - Jika Anda ingin membuat peran skenario lanjutan untuk Amazon Cognito, pilih Amazon Cog nito.

     > **Note**
     >
     > Anda harus membuat peran secara manual untuk digunakan dengan Amazon Cognito hanya ketika Anda mengerjakan skenario lanjutan. Jika tidak, Amazon Cognito dapat membuat peran untuk Anda. Untuk informasi selengkapnya tentang Amazon Cognito, lihat penyedia identitas eksternal kumpulan identitas (identitas federasi) di Panduan Pengembang Amazon Cognito.

   - Jika Anda ingin membuat peran untuk Tind GitHub akan, Anda harus mulai dengan menambahkan penyedia GitHub OIDC ke IAM. Setelah Anda menambahkan penyedia GitHub OIDC ke IAM, pilih token.actions.githubusercontent.com.

     > **Note**
     >
     > Untuk informasi tentang cara mengonfigurasi AWS untuk mempercay GitHub ai penyedia OIDC sebagai identitas federasi, lihat GitHub Dokumen - Mengkonfigurasi OpenID Connect di Amazon Web Services. Untuk informasi tentang praktik terbaik untuk membatasi akses untuk peran yang terkait dengan IdP IAM untuk GitHub, lihat Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC di halaman ini.

- Jika Anda ingin membuat peran untuk HashiCorp Cloud Platform (HCP) Terraform, Anda harus mulai dengan menambahkan penyedia Terraform OIDC ke IAM. Setelah Anda menambahkan penyedia Terraform OIDC ke IAM, pilih app.terraform.io.

  > **Important**
  >
  > Peran IAM untuk HashiCorp Penyedia Terraform OIDC Cloud Platform (HCP) harus mengevaluasi kunci kondisi IAM,`app.terraform.io:sub`, dalam kebijakan kepercayaan peran. Kunci kondisi ini membatasi organisasi HCP Terraform, proyek, ruang kerja, atau fase jalankan yang dapat mengambil peran. Tanpa kunci kondisi ini, kebijakan kepercayaan Anda memberikan akses ke peran dan AWS sumber daya Anda dengan identitas di luar organisasi Anda, yang tidak selaras dengan prinsip hak istimewa paling kecil.
  > Jika Anda menetapkan atau mengubah kebijakan kepercayaan peran untuk peran yang terkait dengan penyedia HCP Terraform OIDC di AWS akun Anda, tetapi tidak mengevaluasi kunci kondisi IAM`app.terraform.io:sub`, Anda akan menerima kesalahan. Selain itu, AWS STS akan menolak permintaan otorisasi jika kebijakan kepercayaan peran Anda tidak mengevaluasi kunci kondisi ini.

5. Informasi yang diminta bervariasi berdasarkan penyedia OIDC yang Anda pilih.

   - Masukkan pengenal untuk aplikasi Anda. Label pengenal berubah berdasarkan penyedia yang Anda pilih:
     - Jika Anda ingin membuat peran untuk Login dengan Amazon, masukkan ID aplikasi ke dalam kotak ID Aplikasi.
     - Jika Anda ingin membuat peran untuk Facebook, masukkan ID aplikasi ke dalam kotak ID Aplikasi.
     - Jika Anda ingin membuat peran untuk Google, masukkan nama audiens ke dalam kotak Audiens.
     - Jika Anda ingin membuat peran untuk Amazon Cognito, masukkan ID kumpulan identitas yang telah Anda buat untuk aplikasi Amazon Cognito ke dalam kotak Identity Pool ID.
   - Jika Anda ingin membuat peran untuk T GitHub indakan, masukkan detail berikut:
     - Untuk Audiens, pilih `sts.amazonaws.com`.
     - Untuk GitHub organisasi, masukkan nama GitHub organisasi. Nama GitHub organisasi diperlukan dan harus alfanumerik termasuk tanda hubung (-). Anda tidak dapat menggunakan karakter wildcard (* dan?) dalam nama GitHub organisasi.

- (Opsional) Untuk GitHub repositori, masukkan nama GitHub repositori. Jika Anda tidak menentukan nilai, nilai default akan menjadi wildcard ()*.
- (Opsional) Untuk GitHub cabang, masukkan nama GitHub cabang. Jika Anda tidak menentukan nilai, nilai default akan menjadi wildcard ()*.

- Jika Anda ingin membuat peran untuk HashiCorp Cloud Platform (HCP) Terraform, masukkan detail berikut:
  - Untuk Audiens, pilih `aws.workload.identity`.
  - Untuk Organisasi, masukkan nama organisasi. Anda dapat menentukan karakter wildcard (*) untuk semua organisasi.
  - Untuk Project, masukkan nama proyek. Anda dapat menentukan karakter wildcard (*) untuk semua proyek.
  - Untuk Ruang Kerja, masukkan nama ruang kerja. Anda dapat menentukan karakter wildcard (*) untuk semua ruang kerja.
  - Untuk Run Phase, masukkan nama fase run. Anda dapat menentukan karakter wildcard (*) untuk semua fase yang dijalankan.

6. (Opsional) Untuk Kondisi (opsional), pilih Tam bah Kondisi untuk membuat kondisi tambahan yang harus dipenuhi sebelum pengguna aplikasi Anda dapat menggunakan izin yang diberikan peran tersebut. Misalnya, Anda dapat menambahkan kondisi yang memberikan akses ke AWS sumber daya hanya untuk ID pengguna IAM tertentu. Anda juga dapat menambahkan kondisi ke kebijakan kepercayaan setelah peran dibuat. Untuk informasi selengkapnya, lihat Memperbarui kebijakan kepercayaan peran.
7. Tinjau informasi OIDC Anda dan kemudian pilih Ber ikutnya.
8. IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin, atau pilih Buat kebijakan untuk membuka tab browser baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Pilih kotak centang di samping kebijakan izin yang ingin dimiliki pengguna OIDC. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.
9. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

   Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.
10. Pilih Berikutnya.

11. Untuk Nama peran, masukkan nama peran. Nama peran harus unik di dalam Anda Akun AWS. Mereka tidak tergantung kasus. Misalnya, Anda tidak dapat membuat peran bernama keduanya **`PRODROLE`** dan**`prodrole`**. Karena AWS sumber daya lain mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah Anda membuatnya.
12. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
13. Untuk mengedit kasus penggunaan dan izin untuk peran, pilih Edit di bagian Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin.
14. (Opsional) Untuk menambahkan metadata ke peran, lampirkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
15. Tinjau peran, lalu pilih Buat peran.

Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC

Jika Anda menggunakan GitHub sebagai penyedia identitas OpenID Connect (OIDC) (IdP), praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IdP IAM. Bila Anda menyertakan pernyataan kondisi dalam kebijakan kepercayaan, Anda dapat membatasi peran untuk GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan kunci kondisi `token.actions.githubusercontent.com:sub` dengan operator kondisi string untuk membatasi akses. Sebaiknya batasi kondisi ke kumpulan repositori atau cabang tertentu dalam GitHub organisasi Anda. Untuk informasi tentang cara mengonfigurasi AWS untuk mempercay GitHub ai OIDC sebagai identitas federasi, lihat GitHub Dokumen - Mengkonfigurasi OpenID Connect di Amazon Web Services.

Jika Anda menggunakan GitHub lingkungan dalam alur kerja tindakan atau dalam kebijakan OIDC, sebaiknya tambahkan aturan perlindungan ke lingkungan untuk keamanan tambahan. Gunakan cabang dan tag penerapan untuk membatasi cabang dan tag mana yang dapat digunakan ke lingkungan. Untuk informasi selengkapnya tentang mengonfigurasi lingkungan dengan aturan perlindungan, lihat Cabang dan tag pener GitHub apan di artikel Menggunakan lingkungan untuk penerapan.

Ketika GitHub IDC IdP adalah Principal tepercaya untuk peran Anda, IAM memeriksa kondisi kebijakan kepercayaan peran untuk memverifikasi bahwa kunci kondisi `token.actions.githubusercontent.com:sub` ada dan nilainya bukan semata-mata karakter wildcard (* dan?) atau nol. IAM melakukan pemeriksaan ini saat kebijakan kepercayaan dibuat atau diperbarui. Jika kunci kondisi tidak `token.actions.githubusercontent.com:sub` ada, atau

nilai kunci tidak memenuhi kriteria nilai yang disebutkan, permintaan akan gagal dan mengembalikan kesalahan.

> **Important**
>
> Jika Anda tidak membatasi kunci kondisi `token.actions.githubusercontent.com:sub` untuk organisasi atau repositori tertentu, Tindakan GitHub dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan IdP GitHub IAM di akun Anda. AWS

Contoh kebijakan kepercayaan berikut membatasi akses ke GitHub organisasi, repositori, dan cabang yang ditentukan. `token.actions.githubusercontent.com:sub`Nilai kunci kondisi dalam contoh berikut adalah format nilai subjek default yang didokumentasikan oleh GitHub.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

Kondisi contoh berikut membatasi akses ke GitHub organisasi dan repositori yang ditentukan, tetapi memberikan akses ke cabang mana pun di dalam repositori.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

Kondisi contoh berikut membatasi akses ke setiap repositori atau cabang dalam GitHub organisasi yang ditentukan. Sebaiknya batasi kunci kondisi `token.actions.githubusercontent.com:sub` ke nilai tertentu yang membatasi akses ke Tind GitHub akan dari dalam GitHub organisasi Anda.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Untuk informasi selengkapnya tentang kunci federasi OIDC yang tersedia untuk pemeriksaan kondisi dalam kebijakan, lihatKunci yang tersedia untuk AWS Federasi OIDC.

Buat peran untuk federasi SAML 2.0 (konsol)

Anda dapat menggunakan federasi SAML 2.0 alih-alih membuat pengguna IAM di federasi Anda. Akun AWS Dengan penyedia identitas (IdP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk mengakses AWS sumber daya di akun Anda. Untuk informasi lebih lanjut tentang federasi dan penyedia identitas, lihat Penyedia identitas dan federasi ke AWS.

> **Note**
>
> Untuk meningkatkan ketahanan federasi, sebaiknya Anda mengonfigurasi IdP dan AWS federasi untuk mendukung beberapa titik akhir masuk SAML. Untuk detailnya, lihat artikel Blog AWS Keamanan Cara menggunakan titik akhir SAML regional untuk failover.

Prasyarat untuk membuat peran untuk SAML

Sebelum Anda dapat membuat peran untuk federasi SAML 2.0, Anda harus terlebih dahulu menyelesaikan langkah-langkah prasyarat berikut.

Untuk mempersiapkan pembuatan peran federasi SAML 2.0

1. Sebelum membuat peran untuk SAML-based federasi, Anda harus membuat penyedia SAML di IAM. Untuk informasi selengkapnya, lihat Buat penyedia identitas SAML di IAM.
2. Persiapkan kebijakan untuk peran yang akan diasumsikan oleh pengguna yang diotentikasi–SAML 2.0. Sebagaimana peran apa pun, peran untuk federasi SAML mencakup dua kebijakan. Salah satunya adalah kebijakan kepercayaan peran yang dapat mengasumsikan peran. Yang lainnya adalah kebijakan izin IAM yang menentukan AWS tindakan dan sumber daya yang diizinkan atau ditolak aksesnya oleh prinsipal federasi SAML.

   Saat membuat kebijakan kepercayaan untuk peran Anda, Anda harus menggunakan tiga nilai untuk memastikan bahwa hanya aplikasi Anda yang dapat mengambil peran tersebut:

   - Untuk elemen `Action`, gunakan tindakan `sts:AssumeRoleWithSAML`.
   - Untuk elemen `Principal`, gunakan string `{"Federated":`*`ARNofIdentityProvider`*`}`. Mengganti *`ARNofIdentityProvider`* dengan ARN penyedia identitas SAML yang Anda buat di Step 1.
   - Untuk `Condition` elemen, gunakan `StringEquals` kondisi untuk menguji apakah `saml:aud` atribut dari respons SAML cocok dengan URL yang ditampilkan browser Anda saat masuk ke konsol. URL titik akhir masuk ini adalah atribut penerima SAML penyedia identitas Anda. Anda dapat menyertakan URL masuk dalam wilayah tertentu. AWS merekomendasikan penggunaan titik akhir Regional alih-alih titik akhir global untuk meningkatkan ketahanan federasi. Untuk daftar *`region-code`* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir

Jika enkripsi SAML diperlukan, URL masuk harus menyertakan pengenal unik yang ditetapkan untuk AWS penyedia SAML Anda. Anda dapat melihat pengidentifikasi unik dengan memilih penyedia identitas di konsol IAM untuk menampilkan halaman detail.

`https://region-code.signin.aws.amazon.com/saml/acs/IdP-ID`

Contoh kebijakan kepercayaan berikut ini dirancang untuk pengguna federasi SAML:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRoleWithSAML",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:saml-provider/PROVIDER-
NAME"
        },
        "Condition": {
            "StringEquals": {
                "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
            }
        }
    }
}
```

Ganti ARN prinsipal dengan ARN aktual untuk penyedia SAML yang Anda buat di IAM. Itu akan memiliki ID akun dan nama penyedia Anda sendiri.

Membuat peran untuk SAML

Setelah Anda menyelesaikan langkah-langkah prasyarat, Anda dapat membuat peran untuk SAML-based federasi.

Untuk membuat peran untuk SAML-based federasi

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran lalu pilih Buat peran.
3. Pilih jenis peran federasi SAML 2.0.
4. Untuk Pilih penyedia SAML, pilih penyedia untuk peran Anda.
5. Pilih metode tingkat akses SAML 2.0.

   - Pilih Izinkan akses terprogram hanya untuk membuat peran yang dapat diasumsikan secara terprogram dari API atau. AWS AWS CLI
   - Pilih Izinkan terprogram dan Konsol Manajemen AWS akses untuk membuat peran yang dapat diasumsikan secara terprogram dan dari. Konsol Manajemen AWS

   Peran yang dibuat oleh keduanya serupa, tetapi peran yang juga dapat diasumsikan dari konsol mencakup suatu kebijakan kepercayaan dengan syarat tertentu. Kondisi itu secara eksplisit memastikan bahwa audiens SAML (`SAML:aud`atribut) disetel ke titik akhir AWS masuk untuk penyedia SAML Anda.
6. Prosedur untuk mendefinisikan atribut bervariasi tergantung pada jenis akses.

   - Jika Anda membuat peran untuk akses terprogram, pilih atribut dari daftar Atribut. Kemudian, di kotak Nilai, masukkan nilai untuk disertakan dalam peran. Hal ini membatasi akses peran kepada pengguna dari penyedia identitas yang memiliki respon otentikasi SAML (pernyataan) mencakup atribut yang Anda tentukan. Anda harus menentukan setidaknya satu atribut untuk memastikan bahwa peran Anda terbatas pada subset pengguna dalam organisasi Anda.
   - Jika Anda membuat peran untuk program dan Konsol Manajemen AWS akses, bagian Sign-in titik akhir menentukan URL yang ditampilkan browser Anda saat masuk ke konsol. Titik akhir ini adalah atribut penerima SAML penyedia identitas Anda, yang dipetakan ke kunci `saml:aud` konteks. Untuk informasi selengkapnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

     a. Pilih titik Non-regional akhir regional atau titik akhir. Sebaiknya gunakan beberapa titik akhir masuk SAML Regional untuk meningkatkan ketahanan federasi.

     b. Untuk Wilayah, pilih wilayah yang didukung penyedia SAML Anda untuk masuk AWS .

c. Agar Sign-in URL menyertakan pengidentifikasi unik, pilih apakah titik akhir masuk menyertakan pengidentifikasi unik yang ditetapkan untuk AWS penyedia identitas SAML Anda. Opsi ini diperlukan untuk pernyataan SAML terenkripsi. Untuk informasi selengkapnya, lihat Federasi SAML 2.0.

7. Untuk menambahkan lebih banyak kondisi terkait atribut ke kebijakan kepercayaan, pilih Kondisi (opsional), pilih kondisi tambahan, dan tentukan nilai.

> **Note**
>
> Daftar ini mencakup atribut SAML yang paling umum digunakan. IAM mendukung atribut tambahan yang dapat Anda gunakan untuk membuat persyaratan. Untuk daftar atribut didukung, lihat Kunci yang Tersedia untuk Federasi SAML. Jika Anda memerlukan syarat untuk atribut SAML yang didukung yang tidak ada dalam daftar, Anda dapat menambahkan syarat tersebut secara manual. Untuk melakukannya, ubah kebijakan kepercayaan setelah Anda membuat peran tersebut.

8. Tinjau informasi kepercayaan SAML 2.0 Anda, lalu pilih Ber ikutnya.
9. IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk kebijakan izin, atau pilih Buat kebijakan untuk membuka tab browser baru dan membuat kebijakan baru dari awal. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM. Setelah Anda membuat kebijakan, tutup tab tersebut dan kembali ke tab asli Anda. Pilih kotak centang di samping kebijakan izin yang ingin dimiliki pengguna federasi SAML. Jika Anda lebih suka, Anda boleh tidak memilih kebijakan saat ini, kemudian melampirkan kebijakan kepada peran di lain waktu. Secara default, peran tidak memiliki izin.
10. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan.

    Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. Pilih kebijakan yang akan digunakan untuk batas izin.
11. Pilih Berikutnya.
12. Pilih Berikutnya: Tinjauan.
13. Untuk Nama peran, masukkan nama peran. Nama peran harus unik di dalam Anda Akun AWS. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`PRODROLE`** dan **`prodrole`**. Karena AWS sumber daya lain mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah dibuat.
14. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.

15. Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin bagian untuk mengedit kasus penggunaan dan izin untuk peran tersebut.
16. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
17. Tinjau peran, lalu pilih Buat peran.

Setelah membuat peran tersebut, Anda menyelesaikan kepercayaan SAML dengan mengonfigurasi perangkat lunak penyedia identitas dengan informasi tentang AWS. Informasi ini mencakup peran yang ingin digunakan oleh pengguna federasi SAML Anda. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara Idp dan AWS. Lihat informasi yang lebih lengkap di Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim.

Membuat peran menggunakan kebijakan kepercayaan khusus

Anda dapat membuat kebijakan kepercayaan khusus untuk mendelegasikan akses dan mengizinkan orang lain melakukan tindakan di situs Anda Akun AWS. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM.

Untuk informasi tentang cara menggunakan peran untuk mendelegasikan izin, lihat Istilah dan konsep peran.

Membuat peran IAM menggunakan kebijakan kepercayaan khusus (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk membuat peran yang dapat diasumsikan oleh pengguna IAM. Misalnya, asumsikan bahwa organisasi Anda memiliki banyak Akun AWS untuk mengisolasi lingkungan pengembangan dari lingkungan produksi. Untuk informasi tingkat tinggi tentang membuat peran yang memungkinkan pengguna di akun pengembangan mengakses sumber daya di akun produksi, lihatContoh skenario menggunakan akun pengembangan dan produksi terpisah.

Untuk membuat peran menggunakan kebijakan kepercayaan khusus (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Pilih jenis peran kebijakan kepercayaan kustom.

4. Di bagian Kebijakan kepercayaan khusus, masukkan atau tempel kebijakan kepercayaan khusus untuk peran tersebut. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM.
5. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.
6. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.

   Buka bagian batas izin dan pilih Gunakan batas izin untuk mengontrol izin peran maksimum. IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda. Pilih kebijakan yang akan digunakan untuk batas izin.
7. Pilih Berikutnya.
8. Untuk Nama peran, tingkat penyesuaian nama peran ditentukan oleh layanan. Jika layanan mendefinisikan nama peran, opsi ini tidak dapat diedit. Dalam kasus lain, layanan mungkin menentukan awalan peran dan memungkinkan Anda untuk memasukkan akhiran opsional. Beberapa layanan memungkinkan Anda untuk menentukan seluruh nama peran Anda.

   Jika memungkinkan, masukkan nama peran atau akhiran nama peran. Nama peran harus unik di dalam diri Anda Akun AWS. Grup tidak dibedakan berdasarkan huruf besar-kecil. Misalnya, Anda tidak dapat membuat peran dengan nama **`PRODROLE`** dan **`prodrole`**. Karena AWS sumber daya lain mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah dibuat.
9. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
10. (Opsional) Pilih Edit di Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin bagian untuk mengedit kebijakan kustom dan izin untuk peran tersebut.
11. (Opsional) Tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
12. Tinjau peran dan kemudian pilih Buat peran.

Contoh kebijakan untuk mendelegasikan akses

Contoh berikut menunjukkan bagaimana Anda dapat mengizinkan atau memberikan Akun AWS akses ke sumber daya di sumber daya lain Akun AWS. Untuk mempelajari cara membuat kebijakan IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

Topik

- Menggunakan peran untuk mendelegasikan akses ke sumber daya orang lain Akun AWS sumber daya
- Menggunakan kebijakan untuk mendelegasikan akses ke layanan
- Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke bucket Amazon S3 di akun lain
- Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke antrean Amazon SQS di akun lainnya
- Tidak dapat mendelegasikan akun ketika akun ditolak aksesnya

Menggunakan peran untuk mendelegasikan akses ke sumber daya orang lain Akun AWS sumber daya

Untuk tutorial yang menunjukkan cara menggunakan peran IAM untuk memberikan pengguna dalam satu akun akses ke AWS sumber daya yang ada di akun lain, lihatTutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM.

> **Important**
>
> Anda dapat menyertakan ARN untuk peran atau pengguna tertentu dalam elemen `Principal` kebijakan kepercayaan peran. Saat Anda menyimpan kebijakan, AWS mengubah ARN menjadi ID utama yang unik. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat kembali peran atau pengguna. Anda biasanya tidak melihat ID ini di konsol, karena juga ada transformasi balik kembali ke ARN ketika kebijakan kepercayaan ditampilkan. Namun, jika Anda menghapus peran atau pengguna, maka hubungan Anda akan rusak. Kebijakan tidak lagi berlaku, bahkan jika Anda membuat ulang pengguna atau peran karena itu tidak sesuai dengan ID prinsipal yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul di konsol karena tidak AWS dapat lagi memetakannya kembali ke ARN. Oleh karena itu, jika Anda menghapus dan membuat ulang pengguna atau peran yang dirujuk di dalam elemen `Principal` kebijakan kepercayaan, Anda harus mengubah peran untuk mengganti ARN. Itu diubah menjadi ID prinsipal baru saat Anda menyimpan kebijakan.

Menggunakan kebijakan untuk mendelegasikan akses ke layanan

Contoh berikut ini menunjukkan kebijakan yang dapat dilampirkan pada sebuah peran. Kebijakan ini memungkinkan dua layanan, Amazon EMR dan AWS Data Pipeline, untuk mengambil peran.

Layanan kemudian dapat melakukan tugas yang diberikan oleh kebijakan izin yang ditetapkan untuk peran tersebut (tidak ditampilkan). Untuk menetapkan beberapa prinsipal layanan, Anda tidak menentukan dua elemen `Service`; Anda hanya dapat memiliki satu. Sebagai gantinya, Anda menggunakan serangkaian dari beberapa prinsipal layanan sebagai nilai elemen `Service` tunggal.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke bucket Amazon S3 di akun lain

Dalam contoh ini, akun A menggunakan kebijakan berbasis sumber daya (kebijakan bucket Amazon S3) untuk memberikan akun B akses penuh ke bucket S3 akun A. Lalu akun B membuat kebijakan pengguna IAM untuk mendelegasikan akses ke bucket akun A kepada salah satu pengguna di akun B.

Kebijakan S3 bucket di akun A mungkin terlihat seperti kebijakan berikut. Dalam contoh ini, bucket S3 akun A diberi nama amzn-s3-demo-bucket, dan nomor akun B adalah 111122223333. Itu tidak menentukan pengguna individual atau pengguna kelompok di akun B, hanya akun itu sendiri.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Atau, akun A dapat menggunakan Amazon S3 Daftar Kontrol Akses (ACLs) untuk memberikan akses kepada akun B ke bucket S3 atau objek tunggal di dalam sebuah bucket. Dalam hal ini, satu-satunya hal yang berubah adalah bagaimana akun A memberikan akses ke akun B. Akun B masih menggunakan kebijakan untuk mendelegasikan akses ke kelompok IAM di akun B, seperti yang dijelaskan di bagian berikutnya dari contoh ini. Untuk informasi selengkapnya tentang mengendalikan akses pada bucket dan objek S3, lihat Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Administrator akun B dapat membuat sampel kebijakan berikut. Kebijakan ini memungkinkan akses baca ke kelompok atau pengguna di akun B. Kebijakan sebelumnya memberikan akses ke akun B. Namun demikian, kelompok individu dan pengguna di akun B tidak dapat mengakses sumber daya sampai suatu kelompok atau pengguna memberikan izin secara jelas ke sumber daya tersebut. Izin dalam kebijakan ini hanya dapat menjadi subset dari izin yang ada di kebijakan lintas akun sebelumnya. Akun B tidak dapat memberikan lebih banyak izin untuk kelompok dan penggunanya daripada yang diberikan akun A ke akun B dalam kebijakan pertama. Dalam kebijakan ini, elemen `Action` secara jelas ditentukan untuk hanya mengizinkan tindakan `List`, dan elemen `Resource` kebijakan ini sesuai dengan `Resource` untuk kebijakan bucket yang diterapkan oleh akun A.

Untuk menerapkan kebijakan ini akun B menggunakan IAM untuk melampirkannya ke pengguna (atau kelompok) yang sesuai di akun B.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Menggunakan kebijakan berbasis sumber daya untuk mendelegasikan akses ke antrean Amazon SQS di akun lainnya

Dalam contoh berikut, akun A memiliki antrean Amazon SQS yang menggunakan kebijakan berbasis sumber daya yang dilampirkan ke antrean untuk memberikan akses antrean ke akun B. Kemudian akun B menggunakan kebijakan kelompok IAM untuk mendelegasikan akses ke kelompok di akun B.

Contoh kebijakan antrean berikut memberi akun B izin untuk melakukan tindakan `SendMessage` dan `ReceiveMessage` pada antrean akun A yang disebut antrean1, tetapi hanya antara tengah hari hingga pukul 15.00 pada 30 November 2014. Nomor akun Akun B adalah 1111-2222-3333. Akun A menggunakan Amazon SQS untuk menerapkan kebijakan ini.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

Kebijakan akun B untuk mendelegasikan akses ke suatu kelompok di akun B dapat terlihat seperti contoh berikut. Akun B menggunakan IAM untuk melampirkan kebijakan ini ke kelompok (atau pengguna).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

Dalam contoh kebijakan pengguna IAM sebelumnya, akun B menggunakan wildcard untuk memberikan akses ke penggunanya ke semua tindakan Amazon SQS pada antrean akun A. Namun, akun B hanya dapat mendelegasikan akses jika akun B telah diberi akses. Kelompok akun B yang memiliki kebijakan kedua dapat mengakses antrean hanya antara tengah hari hingga pukul 15.00 pada 30 November 2014. Pengguna hanya dapat melakukan tindakan `SendMessage` dan `ReceiveMessage`, sebagaimana ditentukan dalam kebijakan antrean Amazon SQS akun A.

Tidak dapat mendelegasikan akun ketika akun ditolak aksesnya

Tidak Akun AWS dapat mendelegasikan akses ke sumber daya akun lain jika akun lain secara eksplisit menolak akses ke akun induk pengguna. Penolakan ini meluas ke para pengguna dalam akun tersebut, baik apakah pengguna sudah memiliki kebijakan yang memberikan akses kepada mereka atau belum.

Sebagai contoh, akun A menyusun kebijakan bucket pada bucket S3 akun A yang secara eksplisit menolak akses akun B ke bucket akun A. Namun, akun B menyusun kebijakan pengguna IAM yang memberi pengguna akun B akses ke bucket akun A. Penolakan secara eksplisit yang diterapkan kepada bucket S3 akun A meluas ke para pengguna di akun B. Itu membatalkan kebijakan pengguna IAM yang memberikan akses ke pengguna di akun B. (Untuk informasi selengkapnya tentang bagaimana izin dievaluasi, lihat Logika evaluasi kebijakan.)

Kebijakan bucket Akun A dapat terlihat seperti kebijakan berikut ini. Dalam contoh ini, bucket S3 akun A diberi nama amzn-s3-demo-bucket, dan nomor akun B adalah 1111-2222-3333. Akun A menggunakan Amazon S3 untuk menerapkan kebijakan ini.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

Penolakan secara eksplisit ini membatalkan kebijakan apa pun dalam akun B yang memberikan izin untuk mengakses bucket S3 dalam akun A.

Buat peran secara otomatis dengan manajer peran

Banyak AWS layanan bergantung pada peran IAM untuk melakukan tindakan atas nama Anda. Manajer peran IAM adalah pengaturan akun opsional yang secara otomatis menyediakan peran tersebut, jadi Anda tidak perlu mengaturnya sendiri. Manajer peran dirancang untuk membantu Anda mulai membangun dengan AWS layanan tanpa konfigurasi IAM manual tambahan.

Manajer peran tersedia di konsol AWS layanan tertentu, tercantum di the section called “AWS konsol layanan yang mendukung manajer peran” halaman ini.

Bagaimana manajer peran bekerja

Ketika manajer peran diaktifkan, ini menyediakan peran yang sesuai dengan layanan dan sumber daya sebagai bagian dari alur kerja konsol yang relevan. Konsol layanan yang didukung akan menunjukkan kapan manajer peran tersedia. Tidak ada tindakan tambahan yang diperlukan untuk menerima peran yang disediakan manajer peran. Misalnya, saat Anda membuat fungsi AWS Lambda di konsol, Lambda menggunakan pengelola peran untuk membuat peran eksekusi fungsi secara default. Untuk menggunakan peran yang berbeda, pilih peran yang ada atau buat peran baru secara manual.

Manajer peran bekerja dengan menerapkan templat peran, definisi kebijakan kepercayaan peran, dan izin untuk layanan dan kasus penggunaan. Untuk mempelajari lebih lanjut tentang template yang digunakan setiap layanan, lihatIkhtisar template peran. Template untuk tugas yang ditentukan

hanya memberikan tindakan yang dibutuhkan tugas. Template untuk pekerjaan terbuka, seperti menjalankan kode Anda sendiri, memberikan akses luas ke AWS API sehingga Anda dapat membangun tanpa memberikan akses ke setiap layanan sendiri.

Peran IAM yang dibuat oleh manajer peran berperilaku seperti peran lain yang Anda buat. Anda dapat melihat, mengedit, atau menghapusnya. Setiap peran juga mencatat template peran yang berasal, sehingga Anda dapat mengidentifikasi peran yang dibuat manajer peran. AWS CloudTrail log ketika manajer peran membuat peran sehingga Anda dapat mengaudit kapan dan bagaimana setiap peran dibuat. Pembuatan peran manajer peran muncul CloudTrail sebagai AcquireRole acara.

Prasyarat

Mengaktifkan manajer peran memerlukan `iam:CreateServiceLinkedRole` izin `iam:PutAccountProperties` dan. Menonaktifkan manajer peran memerlukan izin. `iam:PutAccountProperties` Melihat pengaturan saat ini memerlukan `iam:GetAccountProperties` izin. Kebijakan AWS terkelola `IAMFullAccess` mencakup semua izin ini. Untuk semua izin yang dibutuhkan manajer peran, lihatKelola akses ke manajer peran.

Cara mengaktifkan dan menonaktifkan manajer peran

Anda dapat mengaktifkan atau menonaktifkan manajer peran menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API. Manajer peran dikendalikan oleh properti tingkat akun`RoleManager/Enabled`, yang Anda tetapkan menggunakan API. `PutAccountProperties` Nilai `true` memungkinkan manajer peran; `false` menonaktifkannya.

> **Note**
>
> Untuk sebagian besar AWS akun, manajer peran dinonaktifkan secara default. Tetapi jika Anda membuat akun menggunakan AWS pengalaman baru kami, manajer peran diaktifkan secara default dan tidak dapat dinonaktifkan hingga Anda mengaktifkan fitur-fitur lanjutan. Untuk informasi selengkapnya, lihat Aktifkan AWS fitur lanjutan.

Untuk mengaktifkan atau menonaktifkan manajer peran (konsol)

Manajer peran dapat diaktifkan atau dinonaktifkan pada halaman Pengaturan akun di konsol IAM. Halaman itu juga menunjukkan apakah manajer peran diaktifkan untuk akun tersebut.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM.
2. Di panel navigasi, pilih Pengaturan akun.

3. Di bagian manajer peran, pilih Aktifkan atau Nonaktifkan.

Peran yang dibuat oleh manajer peran tidak akan dihapus saat Anda menonaktifkan manajer peran.

Untuk mengaktifkan atau menonaktifkan manajer peran (AWS CLI)

Setel `RoleManager/Enabled` properti `true` untuk mengaktifkan manajer peran, atau `false` untuk menonaktifkannya.

- Untuk mengaktifkan manajer peran, jalankan`put-account-properties`:

```
aws iam put-account-properties --properties RoleManager/Enabled=true
```

- Untuk menonaktifkan manajer peran, jalankan`put-account-properties`:

```
aws iam put-account-properties --properties RoleManager/Enabled=false
```

- Untuk melihat apakah manajer peran diaktifkan untuk akun, jalankan`get-account-properties`:

```
aws iam get-account-properties
```

Untuk mengaktifkan atau menonaktifkan manajer peran (AWS API)

- Untuk mengaktifkan atau menonaktifkan manajer peran, panggil PutAccountPropertiesdengan `RoleManager/Enabled` properti yang disetel ke `true` atau`false`.
- Untuk melihat pengaturan saat ini, panggil GetAccountProperties.

Untuk informasi selengkapnya, lihat PutAccountPropertiesdan GetAccountPropertiesdi Referensi API IAM.

AWS konsol layanan yang mendukung manajer peran

Manajer peran mendukung pembuatan peran untuk konsol AWS layanan berikut:

- AWS Backup
- AWS CloudFormation
- Amazon CloudWatch
- Amazon EKS

- AWS Elastic Beanstalk
- Amazon EventBridge
- AWS Glue
- AWS Lambda
- Amazon RDS
- Studio SageMaker Terpadu Amazon
- AWS Secrets Manager
- AWS Step Functions

AWS akan terus menambahkan dukungan layanan dari waktu ke waktu. Untuk mempelajari lebih lanjut tentang templat peran yang digunakan setiap layanan, lihatIkhtisar template peran.

Informasi Terkait

- Kelola akses ke manajer peran
- Terapkan izin hak istimewa terkecil ke peran yang dibuat secara otomatis
- Ikhtisar template peran

Kelola akses ke manajer peran

Anda mengontrol akses ke manajer peran di dua tingkat. Izin IAM menentukan siapa di akun yang dapat mengaktifkan manajer peran dan siapa yang dapat menggunakan peran yang disediakannya. Dalam organisasi, kebijakan kontrol layanan (SCP) menentukan apa yang dapat dilakukan akun anggota.

Izin untuk manajer peran

Untuk mengaktifkan atau menonaktifkan manajer peran, Anda memerlukan `iam:PutAccountProperties` izin. Kebijakan yang AWS dikelola `IAMFullAccess` termasuk itu.

Untuk menggunakan pengelola peran, pengguna memerlukan izin IAM untuk tindakan yang dilakukan manajer peran atas nama mereka. Manajer peran menyediakan peran melalui `AcquireRole` API, yang mengevaluasi setiap tindakan IAM yang mendasarinya terhadap izin pengguna:

- Jika peran yang cocok dengan template sudah ada di akun, `AcquireRole` mengembalikan peran itu. Ini membutuhkan `iam:GetRole` izin.

- Jika tidak ada peran yang cocokAcquireRole, buat satu dari template. Ini memerlukan iam:CreateRole izin, ditambah izin untuk apa yang didefinisikan template: iam:PutRolePolicy jika template menambahkan kebijakan sebaris, dan iam:AttachRolePolicy jika templat melampirkan kebijakan terkelola.

Pengguna dengan akses IAM penuh memiliki semua izin yang diperlukan manajer peran. Untuk melihat tindakan tepat yang diperlukan template tertentu, ambil template dengan GetRoleTemplateVersion dan tinjau kebijakannya.

Ketika manajer peran membuat peran, AWS CloudTrail merekam pembuatan sebagai AcquireRole peristiwa tunggal. Acara menunjukkan siapa yang memanggil operasi, template peran dan nilai parameter yang digunakan, dan peran yang dibuat.

Contoh berikut menunjukkan bahwa manajer peran memang AcquireRole gil operasi atas nama pengguna yang mengambil peranAdmin, menggunakan template PowerUserRoleTemplate peran untuk membuat peranPowerUserRole.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123EXAMPLE:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "accountId": "123456789012",
        "accessKeyId": "ASIAWZYMLEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2026-08-11T18:18:50Z",
                "mfaAuthenticated": "false"
            }
        },
        "onBehalfOf": {
            "userId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
            "identityStoreArn": "arn:aws:identitystore::111122223333:identitystore/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    },
    "eventTime": "2026-08-11T18:23:00Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "AcquireRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/151.0.0.0 Safari/537.36",
    "requestParameters": {
        "templateArn": "arn:aws:iam::aws:role-template/iam.amazonaws.com/
PowerUserRoleTemplate:1",
        "templateMinorVersion": 0,
        "replacementValues": {
            "AWSServiceName": {
                "values": [
                    "iam.amazonaws.com"
                ]
            }
        }
    },
    "responseElements": {
        "role": {
            "path": "/",
            "roleName": "PowerUserRole",
            "roleId": "AROA987EXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/PowerUserRole",
            "createDate": "2026-08-11T18:23:00Z",
            "assumeRolePolicyDocument": "%7B%22Version%22:%222012-10-17%22,
%22Statement%22:%5B%7B%22Effect%22:%22Allow%22,%22Principal%22:%7B%22Service%22:
%22iam.amazonaws.com%22%7D,%22Action%22:%22sts:AssumeRole%22%7D%5D%7D"
        }
    },
    "requestID": "fde3dac0-73ac-491c-876b-EXAMPLE89c57",
    "eventID": "c93332cf-1429-4aee-bcaa-EXAMPLE3790d",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
```

```
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

Kontrol peran manajer dalam organisasi

Anda dapat mengontrol manajer peran di seluruh AWS organisasi dengan kebijakan kontrol layanan Organisasi (SCP). Contoh berikut menunjukkan cara memblokir pengelola peran yang mengaktifkan, membuat peran blok, atau memblokir pembuatan peran berbasis template saja.

Contoh-contoh ini menggunakan strategi daftar tol ak, yang berarti Anda juga memerlukan `FullAWSAccess` atau kebijakan lain yang memungkinkan tindakan yang dilampirkan ke entitas organisasi Anda. Anda masih perlu memberikan izin kepada kepala sekolah Anda dengan kebijakan berbasis identitas.

Untuk mencegah akun anggota mengaktifkan manajer peran melalui konsol atau API, tolak `iam:PutAccountProperties` izin tersebut. Kami merekomendasikan kebijakan ini saat Anda membuat akun baru secara langsung melalui Organisasi.

```
{
    "Sid": "DenyRoleManagerEnablement",
    "Effect": "Deny",
    "Action": "iam:PutAccountProperties",
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "iam:AccountPropertyNamespaces": "RoleManager"
        }
    }
}
```

Blokir akses untuk menggunakan manajer peran

Untuk memblokir penggunaan role manager, gunakan kunci `iam:RoleTemplateARN` konteks, yang ada pada tindakan IAM yang membuat peran dari template. Kebijakan berikut menolak semua tindakan IAM yang memiliki nilai non-null untuk kunci `iam:RoleTemplateARN` konteks. Kebijakan ini tidak mencegah semua pembuatan peran — kebijakan ini hanya memblokir pembuatan peran melalui pengelola peran dan pembuatan peran berbasis template melalui SDK.

```
{
    "Sid": "DenyTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": "iam:*",
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": false
        }
    }
}
```

Blokir pembuatan peran kecuali melalui manajer peran

Untuk mengizinkan pembuatan peran hanya melalui manajer peran, gunakan kebijakan berikut untuk memblokir pembuatan peran non-manajer peran. Kebijakan ini masih mengizinkan pembuatan peran berbasis template secara langsung melalui SDK.

```
{
    "Sid": "DenyNonTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": [
        "iam:CreateRole",
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy",
        "iam:PutRolePermissionsBoundary",
        "iam:TagRole"
    ],
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": true
        }
    }
}
```

Informasi Terkait

- Buat peran secara otomatis dengan manajer peran
- Terapkan izin hak istimewa terkecil ke peran yang dibuat secara otomatis
- Ikhtisar template peran

Terapkan izin hak istimewa terkecil ke peran yang dibuat secara otomatis

Saat membuat sumber daya di konsol layanan, manajer peran menyediakan peran IAM yang dibutuhkan sumber daya. Untuk beberapa tugas, seperti menjalankan kode Anda sendiri, AWS tidak dapat menentukan terlebih dahulu izin apa yang diperlukan. Akibatnya, peran dapat mengizinkan lebih banyak izin daripada yang dibutuhkan sumber daya. Setelah mengetahui izin mana yang dibutuhkan sumber daya, Anda dapat mengurangi izin peran agar sesuai.

Menonaktifkan manajer peran tidak memengaruhi peran yang sebelumnya dibuat oleh manajer peran. Anda dapat menonaktifkan pengelola peran tanpa mengurangi izin apa pun, dan Anda dapat mengurangi izin peran tanpa menonaktifkan pengelola peran.

Kapan menerapkan izin hak istimewa terkecil

Memberikan peran hanya izin yang dibutuhkan sumber dayanya adalah praktik terbaik keamanan sejalan dengan prinsip hak istimewa terendah. Untuk sebagian besar peran, manajer peran membuat peran dengan izin cakupan yang baik. Untuk beberapa peran, seperti untuk sumber daya komputasi atau manajemen infrastruktur cloud, ini menciptakan peran dengan izin luas.

Saat Anda siap untuk mulai menelusuri peran yang dibuat oleh manajer peran, sebaiknya mulai dengan peran yang terkait dengan beban kerja dan sumber daya Anda yang paling sensitif, seperti apa pun yang menangani PII atau data produksi. Untuk mempelajari selengkapnya, lihat Bersiaplah untuk izin hak istimewa terkecil.

Cara menganalisis akses yang tidak digunakan

Sebelum menonaktifkan pengelola peran AWS , sarankan Anda menggunakan IAM Access Analyzer untuk membuat penganalisis akses yang tidak digunakan di akun Anda. Analyzer membandingkan tindakan yang diperbolehkan untuk dilakukan peran dengan tindakan yang telah dilakukannya, dan melaporkan izin yang tidak digunakan. Penganalisis meninjau peran IAM di akun Anda, termasuk peran yang Anda buat sendiri.

Di mana temuan muncul tergantung pada bagaimana penganalisis disediakan:

- Penganalisis yang disediakan oleh manajer peran: Saat AWS menyediakan penganalisis akses yang tidak digunakan untuk akun Anda (lihat catatan di bagian ini), temuan akan muncul di halaman Peran di konsol IAM. Untuk setiap peran, Anda dapat melihat jumlah izin yang tidak digunakan dan membuka rekomendasi. Gunakan prosedur berikut untuk mengurangi izin peran dari halaman Per an. Setelah Anda menonaktifkan pengelola peran, halaman pengaturan akun

menyediakan tombol Lihat peran yang membuka halaman Peran, tempat temuan setiap peran muncul.

- Penganalisis yang dibuat sendiri: Saat Anda membuat penganalisis akses yang tidak digunakan sendiri, temuan dan rekomendasi akan muncul di konsol Penganalisis Akses IAM. Pengalaman itu tidak berubah oleh manajer peran. Untuk meninjau dan bertindak atas temuan akses yang tidak digunakan, lihat temuan Penganalisis Akses IAM.

> **Note**
>
> Jika AWS pengelola peran diaktifkan untuk akun Anda yang dibuat menggunakan AWS pengalaman baru, Anda tidak perlu membuat penganalisis saat pertama kali menonaktifkan pengelola peran. Setelah Anda mengaktifkan fitur lanjutan dan menonaktifkan manajer peran, AWS menyediakan penganalisis akses yang tidak digunakan tanpa biaya tambahan selama 90 hari. Penganalisis memberi Anda visibilitas ke dalam izin peran yang tidak digunakan dan rekomendasi cakupan kebijakan sehingga Anda dapat memperbarui peran di akun Anda, termasuk yang dibuat oleh manajer peran, menuju hak istimewa yang paling sedikit.

Prasyarat

Untuk meninjau temuan Access Analyzer, Anda memerlukan izin untuk melakukan `access-analyzer:ListFindings` tindakan. Untuk menerapkan rekomendasi, Anda memerlukan izin untuk melakukan`iam:CreatePolicy`,`iam:AttachRolePolicy`, dan `iam:DetachRolePolicy` pada peran.

Untuk mengurangi izin peran di konsol (penganalisis disediakan oleh manajer peran)

Langkah-langkah ini berlaku jika AWS disediakan penganalisis akses yang tidak digunakan untuk akun Anda. Jika Anda membuat penganalisis sendiri, tinjau dan terapkan rekomendasi di konsol Penganalisis Akses IAM sebagai gantinya. Lihat temuan https://docs.aws.amazon.com/IAM/latest/UserGuide/access-analyzer-findings.html IAM Access Analyzer.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM.
2. Di panel navigasi, pilih Peran.
3. Pada halaman Peran, di kolom Temuan, pilih tautan Izin yang tidak digunakan untuk peran yang izinnya ingin Anda kurangi.

4. Pada halaman Kur angi izin peran, tinjau izin yang tidak digunakan yang dilaporkan untuk peran tersebut.
5. Bandingkan izin yang disarankan dengan izin peran saat ini, dan tinjau tindakan yang dihapus oleh rekomendasi.
6. Pilih Buat perubahan.

Cara kerja izin yang disarankan

Berdasarkan temuan penganalisis akses yang tidak digunakan, IAM membuat satu atau beberapa kebijakan yang dikelola pelanggan yang berisi izin yang disarankan. IAM melampirkan kebijakan baru sebelum melepaskan yang lama, sehingga peran menyimpan izinnya selama perubahan.

Misalnya, pertimbangkan peran eksekusi AWS Lambda. IAM Access Analyzer melaporkan izin peran mana yang tidak digunakan dan merekomendasikan kebijakan yang mencakup tindakan yang dilakukan peran tersebut, seperti menulis log ke Amazon CloudWatch Logs. Anda meninjau rekomendasi terhadap apa yang dilakukan fungsi, mengonfirmasi bahwa itu mencakup semua yang dibutuhkan fungsi, dan menerapkannya. Peran kemudian hanya memungkinkan tindakan yang dibutuhkan fungsi.

> **Important**
>
> Rekomendasi ini didasarkan pada 30 hari terakhir aktivitas. Izin yang jarang digunakan sumber daya pada cakrawala waktu yang lebih lama, seperti untuk pekerjaan triwulanan, dapat muncul sebagai tidak digunakan. Tinjau rekomendasi terhadap kebutuhan beban kerja Anda sebelum menerapkannya.
>
> Setelah Anda mengubah kebijakan peran atau kebijakan kepercayaan, manajer peran tidak lagi mengelola peran tersebut. Peran tidak akan menerima perubahan versi apa pun pada template peran yang awalnya digunakan untuk membuat peran.

IAM membatasi jumlah kebijakan terkelola yang dapat Anda lampirkan ke peran. Jika menerapkan rekomendasi akan melebihi batas itu, konsol memberi tahu Anda dan tidak menerapkan perubahan. Untuk informasi tentang kuota ini, lihatIAM dan AWS STS kuota.

Informasi Terkait

- Buat peran secara otomatis dengan manajer peran
- Kelola akses ke manajer peran

- Ikhtisar template peran

Ikhtisar template peran

Template peran adalah cetak biru peran yang digunakan untuk membuat peran baru. Template menetapkan kebijakan kepercayaan dan izin peran sehingga Anda mendapatkan peran siap pakai untuk layanan tanpa menulis kebijakan sendiri.

Manajer peran menggunakan templat peran yang AWS dikelola, tetapi Anda juga dapat menggunakan templat peran tanpa manajer peran. Halaman ini menjelaskan cara kerja templat peran dan mencantumkan templat untuk setiap layanan.

Cara kerja template peran

Templat peran berisi kebijakan kepercayaan dan izin, serta memiliki Nama Sumber Daya Amazon (ARN). Kebijakan kepercayaan mendefinisikan kepala sekolah mana yang dapat mengambil peran tersebut. Izin menentukan apa yang dapat dilakukan peran. Versi template tidak dapat diubah. Saat AWS memperbarui template, ia menerbitkan versi baru daripada mengubah yang sudah ada.

Templat peran dapat menerima parameter tambahan untuk menyesuaikan dan mencakup properti dan izin peran. Misalnya, template mungkin mengambil peran yang `RoleName` dibuatnya, atau pengenal sumber daya seperti`bucketName`. Saat manajer peran membuat peran di konsol, manajer peran akan menerapkan parameter ke izin dan kebijakan kepercayaan untuk peran tersebut. Saat Anda membuat peran dari template secara langsung dengan IAM API, Anda menyediakan parameter ini.

Untuk membuat peran dari template, gunakan `iam:AcquireRole` API dengan ARN template dan teruskan parameter yang dibutuhkan template.

Definisi template peran dapat berubah antar versi. Untuk melihat kebijakan kepercayaan saat ini dan izin untuk template, ambil kembali dengan `iam:GetRoleTemplateVersion` menggunakan template ARN. AWS merekomendasikan untuk memeriksa template sebelum mengandalkannya, terutama untuk template yang memberikan akses luas.

Bagaimana manajer peran menggunakan templat

Saat membuat sumber daya di konsol layanan yang didukung, manajer peran menyediakan peran dari templat yang cocok dengan kasus layanan dan penggunaan. Manajer peran memilih template dan memasok parameter untuk Anda. Hasilnya adalah peran yang sama dengan yang akan Anda panggil`iam:AcquireRole`. Untuk mengaktifkan manajer peran, lihatKelola akses ke manajer peran.

Direktori template peran

Tabel berikut mencantumkan templat peran untuk setiap layanan yang didukung manajer peran, dengan templat ARN dan parameter yang diperlukan. Gunakan ARN untuk membuat peran dari template, atau untuk meninjau apa yang diberikan manajer peran untuk layanan tersebut. Layanan terdaftar menurut abjad.

Layanan	Template peran	ARN Templat	Parameter
AWS Backup	AWSBackupDefaultServiceRoleTemplate	`arn:aws:iam::aws:role-template/backup.amazonaws.com/AWSBackupDefaultServiceRoleTemplate:1`	AccountID, RoleName
AWS CloudFormation	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon EKS	AmazonEKSAutoClusterRoleTemplate	`arn:aws:iam::aws:role-template/eks.amazonaws.com/AmazonEKSAutoClusterRoleTemplate:1`	RoleName
Amazon EKS	AmazonEKSAutoNodeRoleTemplate	`arn:aws:iam::aws:role-template/ec2.amazonaws.com/AmazonEKSAutoNodeRoleTemplate:1`	RoleName
AWS Elastic Beanstalk	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon EventBridge	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
AWS Glue	AWSGlueCrawlerS3PathsRoleTemplate	`arn:aws:iam::aws:role-template/glue.amazonaws.com/`	RoleName, AccountID, s3path

Layanan	Template peran	ARN Templat	Parameter
		`AWSGlueCrawlerS3PathsRoleTemplate:1`	
AWS Glue	AWSGluePowerUserRoleTemplate	`arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGluePowerUserRoleTemplate:1`	AccountID, RoleName
AWS Lambda	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon RDS	AmazonRDSDirectoryServiceIntegrationRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSDirectoryServiceIntegrationRoleTemplate:1`	RoleName, AccountID
Amazon RDS	AmazonRDSExportToS3RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSExportToS3RoleTemplate:1`	RoleName, AccountID, BucketName
Amazon RDS	AmazonRDSImportFromS3RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSImportFromS3RoleTemplate:1`	CUSTOM_KMS_KEY,, RoleName AccountID, bucketName, kunci, keyRegion, kms, awalan AccountId KeyId
Amazon RDS	AmazonRDSMonitoringRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSMonitoringRoleTemplate:1`	RoleName, AccountID

Layanan	Template peran	ARN Templat	Parameter
Amazon RDS	AmazonRDS ProxyRole Template	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSProxyRoleTemplate:2`	PROXY_WITH_CUSTOM_KEY, PROXY_WITH_DB_CONNECTION, PROXY_WITH_SECRET,, accountID, db, dbUser, KeyID, wilayah, secretID RoleName ResourceId
Amazon RDS	AmazonRDS SecretAccessForDMS RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSSecretAccessForDMSRoleTemplate:1`	CUSTOM_KMS_KEY,, AccountID RoleName, kms, wilayah, secretID KeyId
Studio SageMaker Terpadu Amazon	AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate:1`	CMK_ENABLED,, RoleName AccountID, kunci, keyRegion, kms AccountId KeyId
Studio SageMaker Terpadu Amazon	AmazonSageMakerUserIAMPermissiveExecutionRole Template	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate:1`	RoleName, AccountID

Layanan	Template peran	ARN Templat	Parameter
AWS Secrets Manager	AWSSecretsManagerRotationRoleTemplate	`arn:aws:iam::aws:role-template/secretsmanager.amazonaws.com/AWSSecretsManagerRotationRoleTemplate:1`	ADMIN_RESOURCE_ENABLED, CMK_ENABLED,, AccountID, AdminType, kms, wilayah RoleName, ResourceType KeyArn
Amazon CloudWatch	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName

Informasi Terkait

- Buat peran secara otomatis dengan manajer peran
- Kelola akses ke manajer peran
- Referensi API IAM

Manajemen peran IAM

Sebelum pengguna, aplikasi, atau layanan dapat menggunakan peran yang Anda buat, Anda harus memberikan izin untuk beralih ke peran tersebut. Anda dapat menggunakan kebijakan apa pun yang dilampirkan ke grup atau pengguna untuk memberikan izin yang diperlukan. Bagian ini menjelaskan cara memberikan izin kepada pengguna untuk menggunakan peran. Ini juga menjelaskan bagaimana pengguna dapat beralih ke peran dari Konsol Manajemen AWS, Alat untuk Windows PowerShell, AWS Command Line Interface (AWS CLI) dan `AssumeRole`API.

> **Important**
>
> Saat Anda membuat program peran secara terprogram, alih-alih di konsol IAM, Anda memiliki opsi untuk menambahkan `Path` hingga 512 karakter selain `RoleName`, yang panjangnya

dapat mencapai 64 karakter. Namun, jika Anda bermaksud menggunakan peran dengan fitur Switch Role di Konsol Manajemen AWS, maka gabungan `Path` dan `RoleName` tidak dapat melebihi 64 karakter.

Topik

- Lihat akses peran
- Menghasilkan kebijakan berdasarkan informasi akses
- Berikan izin pengguna untuk beralih peran
- Memberikan izin pengguna untuk meneruskan peran ke AWS layanan
- Mencabut kredensi keamanan sementara peran IAM
- Memperbarui peran terkait layanan
- Memperbarui kebijakan kepercayaan peran
- Memperbarui izin untuk peran
- Perbarui setelan untuk peran
- Hapus peran atau profil contoh

Lihat akses peran

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghasilkan kebijakan berdasarkan informasi akses

Terkadang Anda dapat memberikan izin ke entitas IAM (pengguna atau peran) di luar yang mereka butuhkan. Untuk membantu Anda menyaring izin yang Anda berikan, Anda dapat membuat kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas. IAM Access Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam rentang tanggal yang ditentukan. Anda dapat menggunakan templat untuk membuat kebijakan terkelola dengan izin berbutir halus, lalu melampirkannya ke entitas IAM. Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat Pembuatan kebijakan IAM Access Analyzer.

Berikan izin pengguna untuk beralih peran

Ketika administrator membuat peran untuk akses lintas akun, mereka membangun kepercayaan antara akun yang memiliki peran, sumber daya (akun yang dipercaya), dan akun yang berisi pengguna (akun tepercaya). Untuk melakukannya, administrator akun terpercaya menentukan nomor akun terpercaya sebagai `Principal` dalam kebijakan kepercayaan peran tersebut. Itu berpotensi memungkinkan setiap pengguna di akun tepercaya untuk mengambil peran tersebut. Untuk menyelesaikan konfigurasi, administrator akun terpercaya harus memberikan izin kepada kelompok atau pengguna tertentu dalam izin akun tersebut untuk beralih ke peran.

Untuk memberikan izin untuk beralih ke peran

1. Sebagai administrator akun tepercaya, buat kebijakan baru untuk pengguna, atau edit kebijakan yang ada untuk menambahkan elemen yang diperlukan. Lihat perinciannya di Membuat atau mengubah.
2. Kemudian, pilih cara Anda ingin berbagi informasi peran:

   - Tautan peran: Kirim tautan kepada pengguna yang membawa mereka ke halaman Beralih Peran dengan semua detail yang sudah diisi.
   - ID Akun atau alias: Berikan setiap pengguna nama peran bersama dengan nomor ID akun atau alias akun. Pengguna kemudian masuk ke halaman Ganti Peran dan menambahkan detail secara manual.

   Lihat perinciannya di Memberikan informasi kepada pengguna.

Perhatikan bahwa Anda dapat mengganti peran hanya saat Anda masuk sebagai pengguna IAM, peran, atau SAML-federated peran federasi identitas web. Anda tidak dapat beralih peran saat masuk sebagai Pengguna root akun AWS.

> **Important**
>
> Anda tidak dapat mengalihkan peran Konsol Manajemen AWS ke peran yang membutuhkan ExternalId nilai. Anda hanya dapat beralih ke peran tersebut dengan memanggil `AssumeRole` API yang mendukung parameter `ExternalId`.

Catatan

- Topik ini membahas kebijakan untuk pengguna, karena Anda pada akhirnya memberikan izin kepada pengguna untuk menyelesaikan tugas. Namun, kami tidak menyarankan Anda memberikan izin langsung kepada pengguna individual. Ketika pengguna mengambil peran, mereka diberi izin yang terkait dengan peran itu.
- Saat Anda mengganti peran di Konsol Manajemen AWS, konsol selalu menggunakan kredenSIAL asli Anda untuk mengotorisasi sakelar. Ini berlaku apakah Anda masuk sebagai pengguna IAM, sebagai SAML-federated peran, atau sebagai peran federasi identitas web. Misalnya, jika Anda beralih ke rolEA, IAM menggunakan kredentif peran pengguna asli atau federasi untuk menentukan apakah Anda diizinkan untuk mengasumsikan rolEA. Jika Anda kemudian mencoba untuk beralih ke RoleB saat menggunakan RoleA, pengguna asli atau kredensial peran federasi digunakan untuk mengotorisasi upaya Anda. Kredensial untuk RoleA tidak digunakan untuk tindakan ini.

Topik

- Membuat atau mengubah
- Memberikan informasi kepada pengguna

Membuat atau mengubah

Kebijakan yang memberikan izin kepada pengguna untuk mengasumsikan peran harus mencantumkan pernyataan dengan efek `Allow` pada hal berikut:

- Tindakan `sts:AssumeRole`
- Amazon Resource Name (ARN) peran dalam elemen `Resource`

Pengguna yang mendapatkan kebijakan diizinkan untuk beralih peran pada sumber daya yang terdaftar (baik melalui keanggotaan grup atau dilampirkan langsung).

Note

Jika `Resource` diatur menjadi `*`, pengguna dapat mengasumsikan peran apa pun dalam akun apa pun yang memercayai akun pengguna. (Dengan kata lain, kebijakan kepercayaan peran menentukan akun pengguna sebagai `Principal`). Sebaiknya, kami menyarankan

agar Anda mengikuti prinsip hak istimewa terendah dan menentukan ARN lengkap hanya untuk peran yang dibutuhkan pengguna.

Contoh berikut menunjukkan kebijakan yang memungkinkan pengguna untuk mengasumsikan peran hanya dalam satu akun. Selain itu, kebijakan ini menggunakan wildcard (*) untuk menentukan bahwa pengguna dapat beralih ke peran hanya jika nama peran dimulai dan huruf `Test`.

JSON

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

Note

Izin yang diberikan oleh peran kepada pengguna tidak menambah izin yang sudah diberikan kepada pengguna. Saat pengguna beralih ke suatu peran, pengguna sementara waktu akan memberikan izin awalnya sebagai ganti atas apa yang diberikan oleh peran tersebut. Saat pengguna keluar dari peran, izin pengguna asli mereka dipulihkan secara otomatis. Misalnya, katakanlah izin pengguna memungkinkan bekerja dengan instans Amazon EC2, tetapi kebijakan izin peran tidak memberikan izin tersebut. Dalam hal ini, saat menggunakan peran tersebut, pengguna tidak dapat bekerja dengan instans Amazon EC2 di dalam konsol. Selain itu, kredensial sementara yang diperoleh melalui `AssumeRole` tidak bekerja dengan instans Amazon EC2 secara terprogram.

Memberikan informasi kepada pengguna

Setelah Anda membuat peran dan memberikan izin kepada pengguna Anda untuk beralih ke sana, Anda harus memberikan hal-hal berikut kepada pengguna:

- Nama peran

- ID atau alias akun yang berisi peran tersebut

Anda dapat merampingkan akses untuk pengguna Anda dengan mengirimkan tautan yang telah dikonfigurasi sebelumnya dengan ID akun dan nama peran. Anda dapat melihat tautan peran setelah menyelesaikan panduan Buat Peran dengan memilih spanduk Lihat Peran, atau pada halaman Ringkasan Per an untuk setiap peran yang diaktifkan lintas akun.

Anda juga dapat menggunakan format berikut untuk menyusun tautan secara manual. Ganti ID akun atau alias Anda dan nama peran untuk dua parameter dalam contoh berikut.

`https://signin.aws.amazon.com/switchrole?account=`*`your_account_ID_or_alias`*`&roleName=`*`optional_path/role_name`*

Kami menyarankan Anda mengarahkan pengguna Anda Beralih dari pengguna ke peran IAM (konsol) untuk memandu mereka melalui proses. Untuk memecahkan masalah umum yang mungkin Anda temui saat mengasumsikan peran, lihat Saya tidak dapat mengsumsikan peran.

Pertimbangan-pertimbangan

- Jika Anda membuat peran secara terprogram, Anda dapat membuat peran dengan jalur dan nama. Jika demikian, Anda harus memberikan jalur lengkap dan nama peran kepada pengguna agar mereka dapat masuk ke halaman Beralih Peran di Konsol Manajemen AWS. Sebagai contoh: `division_abc/subdivision_efg/role_XYZ`.
- Jika Anda membuat peran secara terprogram, Anda dapat menambahkan `Path` hingga 512 karakter dan a. `RoleName` Panjang nama peran dapat mencapai 64 karakter. Namun, untuk menggunakan peran dengan fitur Ber alih Peran di Konsol Manajemen AWS, gabungan `Path` dan `RoleName` tidak boleh melebihi 64 karakter.

- Untuk tujuan keamanan, Anda dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan di AWS. Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan identias saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta agar pengguna IAM menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS. Untuk informasi selengkapnya, lihat `sts:SourceIdentity`. Anda juga dapat menggunakan `sts:RoleSessionName` untuk mengharuskan pengguna menentukan nama sesi saat mereka mengambil peran. Hal ini dapat membantu Anda membedakan antara sesi peran ketika peran digunakan oleh prinsipal yang berbeda.

Memberikan izin pengguna untuk meneruskan peran ke AWS layanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus mener uskan peran IAM ke layanan. Ini memungkinkan layanan untuk mengambil peran nanti dan melakukan tindakan atas nama Anda. Untuk sebagian besar layanan, Anda hanya perlu meneruskan peran ke layanan satu kali selama penyiapan, dan tidak setiap kali layanan mengambil peran tersebut. Misalnya, anggap Anda memiliki aplikasi yang berjalan di instans Amazon EC2 Aplikasi tersebut memerlukan kredensial sementara untuk autentikasi, dan izin untuk mengotorisasi aplikasi guna melakukan tindakan di AWS. Saat menyiapkan aplikasi, Anda harus meneruskan peran ke Amazon EC2 untuk digunakan dengan instans yang menyediakan kredenSIAL tersebut. Anda menentukan izin untuk aplikasi yang berjalan pada instans dengan melampirkan kebijakan IAM pada peran tersebut. Aplikasi ini mengasumsikan peran setiap kali diperlukan untuk melakukan tindakan yang diizinkan oleh peran tersebut.

Untuk meneruskan peran (dan izinnya) ke AWS layanan, pengguna harus memiliki izin untuk meneruskan peran ke layanan. Ini membantu administrator untuk memastikan bahwa hanya pengguna yang disetujui yang dapat mengonfigurasi layanan dengan peran yang memberikan izin. Untuk mengizinkan pengguna meneruskan peran ke AWS layanan, Anda harus memberikan `PassRole` izin kepada pengguna, peran, atau grup IAM pengguna.

> **Warning**
>
> - Anda hanya dapat menggunakan `PassRole` izin untuk meneruskan peran IAM ke layanan yang berbagi AWS akun yang sama. Untuk meneruskan peran dalam Akun A ke layanan di Akun B, Anda harus terlebih dahulu membuat peran IAM di Akun B yang dapat mengambil peran dari Akun A, dan kemudian peran di Akun B dapat diteruskan ke layanan. Lihat perinciannya di Akses sumber daya lintas akun di IAM.
> - Jangan mencoba mengontrol siapa yang dapat melewati peran dengan menandai peran dan kemudian menggunakan kunci `ResourceTag` kondisi dalam kebijakan dengan `iam:PassRole` tindakan tersebut. Pendekatan ini tidak memiliki hasil yang dapat diandalkan.

Saat menyet `PassRole` el izin, Anda harus memastikan bahwa pengguna tidak melewati peran di mana peran tersebut memiliki lebih banyak izin daripada yang Anda inginkan untuk dimiliki pengguna. Misalnya, Alice mungkin tidak diizinkan untuk melakukan tindakan Amazon S3 apa pun. Jika Alice dapat meneruskan peran ke layanan yang memungkinkan tindakan Amazon S3, layanan dapat melakukan tindakan Amazon S3 atas nama Alice saat menjalankan pekerjaan.

Saat Anda menentukan peran terkait layanan, Anda juga harus memiliki izin untuk meneruskan peran tersebut ke layanan. Beberapa layanan secara otomatis membuat peran yang terkait dengan layanan di akun Anda ketika Anda melakukan tindakan di layanan tersebut. Misalnya, Amazon EC2 Auto Scaling membuat peran `AWSServiceRoleForAutoScaling` terkait layanan untuk Anda saat Anda membuat grup Penskalaan Otomatis untuk pertama kalinya. Jika Anda mencoba menentukan peran terkait layanan saat membuat grup Penskalaan Otomatis dan Anda tidak memiliki `iam:PassRole` izin, Anda menerima kesalahan. Jika Anda tidak secara eksplisit menentukan peran, `iam:PassRole` izin tidak diperlukan, dan defaultnya adalah menggunakan `AWSServiceRoleForAutoScaling` peran untuk semua operasi yang dilakukan pada grup tersebut. Untuk mempelajari layanan yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari layanan mana yang secara otomatis membuat peran yang terkait dengan layanan saat Anda melakukan tindakan dalam layanan tersebut, pilih tautan Ya dan lihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Pengguna dapat meneruskan sebuah peran ARN sebagai sebuah parameter dalam setiap operasi API yang menggunakan peran tersebut untuk menetapkan izin ke layanan. Layanan kemudian memeriksa apakah pengguna memiliki izin `iam:PassRole`. Untuk membatasi pengguna agar hanya meneruskan peran yang disetujui, Anda dapat menyaring izin `iam:PassRole` dengan elemen `Resources` pernyataan kebijakan IAM.

Anda dapat menggunakan `Condition` elemen dalam kebijakan JSON untuk menguji nilai kunci yang disertakan dalam konteks permintaan semua AWS permintaan. Untuk mempelajari penggunaan kunci syarat dalam kebijakan, lihat Elemen kebijakan IAM JSON: Kondisi. Kunci syarat `iam:PassedToService` dapat digunakan untuk menentukan ke prinsipal layanan mana sebuah peran dapat diberikan. Untuk mempelajari lebih lanjut tentang menggunakan kunci `iam:PassedToService` kondisi dalam kebijakan, lihat iam: PassedToService.

Contoh 1

Misalkan Anda ingin memberi pengguna kemampuan untuk meneruskan salah satu set peran yang disetujui ke layanan Amazon EC2 saat meluncurkan instance. Anda memerlukan tiga elemen:

- Kebijakan izin IAM yang dilamprkan ke peran yang menentukan apa yang dapat dilakukan peran tersebut. Cakupan diperbolehkan hanya untuk tindakan yang harus dilakukan peran, dan hanya untuk sumber daya yang diperlukan peran untuk tindakan tersebut. Anda dapat menggunakan kebijakan izin IAM yang AWS dikelola atau dibuat pelanggan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Kebijakan kepercayaan untuk peran yang memungkinkan layanan untuk mengasumsikan peran. Misalnya, Anda dapat melampirkan kebijakan kepercayaan berikut pada peran dengan tindakan `UpdateAssumeRolePolicy`. Kebijakan kepercayaan ini memungkinkan Amazon EC2 untuk menggunakan peran dan izin yang dilampirkan pada peran tersebut.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid":
 "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Kebijakan izin IAM yang dil ampirkan ke pengguna IAM yang memungkinkan pengguna untuk hanya meneruskan peran yang disetujui. Anda biasanya `iam:GetRole` menambahkan `iam:PassRole` sehingga pengguna bisa mendapatkan detail peran yang akan dilewatkan. Dalam contoh ini, pengguna hanya dapat meneruskan peran yang ada di akun yang ditentukan dengan nama yang dimulai dengan`EC2-roles-for-XYZ-`:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/EC2-roles-for-XYZ-*"
        }
    ]
}
```

Sekarang pengguna dapat memulai instans Amazon EC2 dengan peran yang ditetapkan. Aplikasi yang berjalan pada instans dapat mengakses kredensial sementara untuk peran tersebut melalui metadata profil instans. Kebijakan izin IAM yang dilamprkan ke peran yang menentukan apa yang dapat dilakukan oleh instans tersebut.

Contoh 2

Amazon Relational Database Service (Amazon RDS) mendukung fitur yang disebut Enhanced Monitoring. Fitur ini memungkinkan Amazon RDS untuk memantau instans basis data dengan menggunakan agen. Ini juga memungkinkan Amazon RDS untuk mencatat metrik ke Amazon CloudWatch Logs. Untuk mengaktifkan fitur ini, Anda harus membuat peran layanan untuk memberi Amazon RDS izin untuk memantau dan menyusun metrik ke log Anda.

Untuk membuat peran untuk pemantauan yang ditingkatkan Amazon RDS.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Peran, lalu pilih Buat peran.
3. Pilih jenis peran AWS Layanan, lalu untuk Kas us penggunaan untuk lainnya Layanan AWS, pilih layanan RDS. Pilih RDS — Pemantauan yang Diting katkan, lalu pilih Ber ikutnya.
4. Pilih kebijakan AmazonRDSEnhancedMonitoringRole izin.
5. Pilih Berikutnya.
6. Untuk Nama peran, masukkan nama peran yang membantu Anda mengidentifikasi tujuan peran ini. Nama peran harus unik di dalam Anda Akun AWS. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dari ARN, nama peran tersebut peka huruf besar/kecil. Ketika nama peran muncul kepada pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil. Karena berbagai entitas mungkin mereferensikan peran, Anda tidak dapat mengedit nama peran setelah dibuat.

7. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran baru ini.
8. (Opsional) Tambahkan metadata ke pengguna dengan cara melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
9. Tinjau peran dan kemudian pilih Buat peran.

Peran tersebut secara otomatis mendapatkan kebijakan kepercayaan yang memberikan izin layanan `monitoring.rds.amazonaws.com` untuk memegang peran tersebut. Setelah itu, Amazon RDS dapat melakukan semua tindakan yang diizinkan oleh kebijakan `AmazonRDSEnhancedMonitoringRole`.

Pengguna yang ingin Anda akses Pemantauan yang Ditingkatkan memerlukan kebijakan yang menyertakan pernyataan yang memungkinkan pengguna untuk mencantumkan peran RDS dan pernyataan yang memungkinkan pengguna untuk meneruskan peran, seperti berikut ini. Gunakan nomor akun Anda dan ganti nama peran dengan nama yang Anda berikan pada langkah 6.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

Anda dapat menggabungkan pernyataan ini dengan pernyataan dalam kebijakan lain atau menempatkannya ke dalam kebijakannya sendiri. Sebagai gantinya, untuk menentukan bahwa pengguna dapat meneruskan peran apa pun yang dimulai dengan`RDS-`, Anda dapat mengganti nama peran dalam sumber daya ARN dengan wildcard, sebagai berikut.

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

`iam: PassRole` tindakan di AWS CloudTrail log

`PassRole`bukan panggilan API. `PassRole`adalah izin, artinya tidak ada CloudTrail log yang dihasilkan untuk IAM`PassRole`. Untuk meninjau peran apa yang diteruskan ke mana Layanan AWS CloudTrail, Anda harus meninjau CloudTrail log yang membuat atau mengubah sumber AWS daya yang menerima peran tersebut. Misalnya, peran diteruskan ke AWS Lambda fungsi saat dibuat. Log untuk `CreateFunction` tindakan menunjukkan catatan peran yang diteruskan ke fungsi.

Mencabut kredensi keamanan sementara peran IAM

> **Warning**
>
> Jika Anda mengikuti langkah-langkah di halaman ini, semua pengguna dengan sesi saat ini yang dibuat dengan asumsi peran ditolak akses ke semua AWS tindakan dan sumber daya. Hal ini dapat mengakibatkan pengguna kehilangan pekerjaan yang belum disimpan.

Ketika Anda mengizinkan pengguna untuk mengakses Konsol Manajemen AWS dengan waktu durasi sesi yang lama (seperti 12 jam), kredensi sementara mereka tidak kedaluwarsa dengan cepat. Jika pengguna secara tidak sengaja mengekspos kredensialnya kepada pihak ketiga yang tidak sah, pihak tersebut memiliki akses selama sesi berlangsung. Namun, Anda dapat segera mencabut semua izin untuk kredensial peran yang diterbitkan sebelum waktu tertentu jika perlu. Semua kredensial sementara untuk peran tersebut yang diterbitkan sebelum waktu yang ditentukan menjadi tidak berlaku. Ini memaksa semua pengguna untuk mengautentikasi ulang dan meminta kredensi baru.

> **Note**
>
> Anda tidak dapat mencabut sesi untuk peran yang terkait dengan layanan.

Saat Anda mencabut izin untuk peran menggunakan prosedur dalam topik ini, AWS lampirkan kebijakan sebaris baru ke peran yang menolak semua izin untuk semua tindakan. Ini mencakup sebuah syarat yang menerapkan pembatasan hanya jika pengguna mengasumsikan peran tersebut sebelum waktu saat Anda mencabut izin. Jika pengguna mengasumsikan peran tersebut setelah Anda mencabut izin, maka kebijakan penolakan tidak berlaku bagi pengguna tersebut.

Untuk informasi lebih lanjut tentang menolak akses, lihatMenonaktifkan izin untuk kredensial keamanan sementara.

> **Important**
>
> Kebijakan penolakan ini berlaku untuk semua pengguna peran yang ditentukan, bukan hanya untuk mereka yang memiliki sesi konsol durasi yang lebih lama.

Izin minimum untuk mencabut izin sesi dari peran

Untuk dapat berhasil mencabut izin sesi dari peran, Anda harus memiliki izin `PutRolePolicy` untuk peran tersebut. Ini memungkinkan Anda untuk melampirkan kebijakan inline `AWSRevokeOlderSessions` ke peran tersebut.

Mencabut izin sesi

Anda dapat mencabut izin sesi dari peran untuk menolak semua izin kepada pengguna mana pun yang mengambil peran tersebut.

> **Note**
>
> Anda tidak dapat mengedit peran di IAM yang dibuat dari kumpulan izin Pusat Identitas IAM. Anda harus mencabut sesi set izin aktif untuk pengguna di Pusat Identitas IAM. Untuk informasi selengkapnya, lihat Mencabut sesi peran IAM aktif yang dibuat oleh set izin di Panduan Pengguna Pusat Identitas IAM.

Untuk segera menolak semua izin untuk pengguna kredensial peran saat ini

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran, lalu pilih nama (bukan kotak centang) peran yang izinnya ingin dicabut.
3. Di halaman Ringkasan untuk peran yang dipilih, pilih tab Cabut sesi.
4. Di tab Cabut sesi, pilih Cabut sesi aktif.
5. AWS meminta Anda untuk mengkonfirmasi tindakan. Pilih Saya mengakui bahwa saya mencabut semua sesi aktif untuk peran ini. kotak centang dan pilih Cabut sesi aktif pada kotak dialog.

IAM kemudian melampirkan kebijakan yang dinamai `AWSRevokeOlderSessions` untuk peran tersebut. Setelah Anda memilih Cabut sesi aktif, kebijakan menolak semua akses ke pengguna yang mengambil peran di masa lalu serta sekitar 30 detik ke depan. Pilihan future time ini memperhitungkan penundaan propagasi kebijakan untuk menangani sesi baru yang diperoleh atau diperbarui sebelum kebijakan yang diperbarui berlaku di wilayah tertentu. Setiap pengguna yang mengambil peran lebih dari sekitar 30 detik setelah Anda memilih Mencabut sesi aktif tidak terpengaruh. Untuk mempelajari mengapa perubahan tidak selalu langsung terlihat, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

> **Note**
>
> Jika Anda memilih untuk Mencabut sesi aktif lagi nanti, stempel tanggal dan waktu dalam kebijakan akan di-refresh dan sekali lagi menolak semua izin untuk setiap pengguna yang mengambil peran sebelum waktu yang ditentukan.

Pengguna valid yang sesinya dicabut dengan cara ini harus memperoleh kredensial sementara untuk sesi baru untuk melanjutkan bekerja. AWS CLI Cache kredensialnya sampai kedaluwarsa. Untuk memaksa CLI menghapus dan menyegarkan kredensial yang tersimpan yang sudah tidak valid, jalankan salah satu perintah berikut:

Linux, macOS, atau Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Mencabut izin sesi sebelum waktu yang ditentukan

Anda juga dapat mencabut izin sesi kapan saja sesuai pilihan Anda menggunakan AWS CLI atau SDK untuk menentukan nilai `aw: TokenIssueTime` kunci dalam elemen Kondisi kebijakan.

Kebijakan ini menolak semua izin jika nilainya lebih awal dari `aws:TokenIssueTime` tanggal dan waktu yang ditentukan. Nilai dari `aws:TokenIssueTime` sesuai dengan waktu yang tepat saat kredensial keamanan sementara dibuat. `aws:TokenIssueTime`Nilai hanya ada dalam konteks

AWS permintaan yang ditandatangani dengan kredenal keamanan sementara, sehingga pernyataan Deny dalam kebijakan tidak memengaruhi permintaan yang ditandatangani dengan kredensi jangka panjang pengguna IAM.

Kebijakan ini juga dapat dilampirkan pada suatu peran. Dalam hal ini, kebijakan hanya memengaruhi kredensial keamanan sementara yang dibuat oleh peran sebelum tanggal dan waktu yang ditentukan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Pengguna valid yang sesinya dicabut dengan cara ini harus memperoleh kredensial sementara untuk sesi baru untuk melanjutkan bekerja. AWS CLI Cache kredensialnya sampai kedaluwarsa. Untuk memaksa CLI menghapus dan menyegarkan kredensial yang tersimpan yang sudah tidak valid, jalankan salah satu perintah berikut:

Linux, macOS, atau Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Memperbarui peran terkait layanan

Metode yang Anda gunakan untuk mengedit peran yang ditautkan ke layanan bergantung pada layanan. Beberapa layanan mungkin mengizinkan Anda mengedit izin untuk peran tertaut-layanan

dari konsol layanan, API, atau CLI. Namun, setelah Anda membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Anda dapat mengedit deskripsi peran apa pun dari konsol IAM, API, atau CLI.

Untuk informasi tentang layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait-Layanan Untuk mempelajari apakah layanan mendukung pengeditan peran terkait layanan, pilih tautan Yes untuk melihat dokumentasi peran terkait layanan untuk layanan itu.

Mengedit deskripsi peran yang ditautkan ke layanan (konsol)

Anda dapat menggunakan konsol IAM untuk menyunting penjelasan peran terkait layanan.

Untuk menyunting deskripsi peran terkait layanan (konsol IAM)

1. Di panel navigasi konsol IAM, pilih Peran.
2. Pilih nama peran yang akan diubah.
3. Di ujung kanan Deskripsi peran, pilih Edit.
4. Masukkan deskripsi baru di kotak, lalu pilih Simpan.

Menyunting deskripsi peran terkait layanan (AWS CLI)

Anda dapat menggunakan perintah IAM dari AWS CLI untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan (AWS CLI)

1. (Opsional) Untuk melihat deskripsi peran saat ini, jalankan perintah berikut:

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Gunakan nama peran, bukan ARN, untuk mereferensikan ke peran dengan perintah CLI. Misalnya, jika peran memiliki ARN berikut: `arn:aws:iam::123456789012:role/myrole`, referensi Anda ke peran sebagai **`myrole`**.
2. Untuk memperbarui deskripsi peran terkait layanan, jalankan perintah berikut:

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

Mengedit deskripsi peran terkait layanan (API)AWS

Anda dapat menggunakan AWS API untuk mengedit deskripsi peran terkait layanan.

Untuk mengubah deskripsi peran terkait layanan (API)AWS

1. (Opsional) Untuk melihat deskripsi peran saat ini, panggil operasi berikut, dan sebutkan nama peran:

   AWS API: GetRole
2. Untuk memperbarui deskripsi peran, hubungi operasi berikut, dan sebutkan nama (dan deskripsi opsional) peran tersebut:

   AWS API: UpdateRole

Memperbarui kebijakan kepercayaan peran

Untuk mengubah siapa yang dapat mengasumsikan peran, Anda harus mengubah kebijakan kepercayaan peran tersebut. Anda tidak dapat mengubah kebijakan kepercayaan untuk peran yang terkait dengan layanan.

> **Catatan**
>
> - Jika pengguna terdaftar sebagai prinsipal dalam kebijakan kepercayaan peran, tetapi tidak dapat mengasumsikan peran tersebut, periksa batas izin pengguna tersebut. Jika batas izin diatur untuk pengguna tersebut, maka itu seharusnya memungkinkan tindakan `sts:AssumeRole`.
> - Untuk mengizinkan pengguna mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti Amazon EC2, Amazon ECS, Amazon EKS, dan Lambda memberikan kredenSIAL sementara dan secara otomatis memperbarui kredenSIAL ini. Ini memastikan bahwa Anda selalu memiliki kumpulan kredenSIAL yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredenSIAL sementara. Namun, jika Anda ingin mener uskan tag sesi atau kebijakan sesi, Anda harus mengambil peran saat ini lagi.

Memperbarui kebijakan kepercayaan peran (konsol)

Untuk mengubah kebijakan kepercayaan peran di Konsol Manajemen AWS

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran.
3. Dalam daftar peran di akun Anda, pilih nama peran yang ingin Anda ubah.
4. Pilih tab Relasi kepercayaan, lalu pilih Edit kebijakan kepercayaan.
5. Ubah kebijakan kepercayaan sebagaimana diperlukan. Untuk menambahkan prinsipal tambahan yang dapat mengasumsikan peran, tentukan dalam elemen `Principal`. Misalnya, cuplikan kebijakan berikut menunjukkan cara mereferensikan dua Akun AWS di `Principal` elemen:

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
  ]
},
```

Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

Cuplikan kebijakan berikut menunjukkan cara mereferensikan dua AWS layanan dalam `Principal` elemen:

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Setelah selesai mengedit kebijakan kepercayaan, pilih Per barui kebijakan untuk menyimpan perubahan.

   Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan Referensi elemen kebijakan IAM JSON.

Untuk mengizinkan pengguna dalam akun eksternal terpercaya untuk menggunakan peran (konsol)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Masuk ke eksternal tepercaya Akun AWS.
2. Tentukan apakah akan melampirkan izin ke pengguna atau ke kelompok. Di panel navigasi konsol IAM, pilih Pengguna atau Grup pengguna yang sesuai.
3. Pilih nama pengguna atau kelompok yang ingin Anda beri akses, dan kemudian pilih tab Izin.
4. Lakukan salah satu langkah berikut:

   - Untuk mengubah kebijakan yang dikelola pelanggan, pilih nama kebijakan tersebut, pilih Ubah Kebijakan, lalu pilih tab JSON. Anda tidak dapat mengedit kebijakan AWS terkelola. AWS kebijakan terkelola muncul dengan AWS ikon (). Untuk informasi selengkapnya tentang perbedaan antara kebijakan yang dikelola AWS dan kebijakan yang dikelola oleh pelanggan, lihat Kebijakan terkelola dan kebijakan inline.
   - Untuk mengubah kebijakan inline, pilih panah di samping nama kebijakan dan pilih Ubah Kebijakan.
5. Dalam editor kebijakan, tambahkan elemen `Statement` baru yang menentukan hal berikut:

   ```
   {
     "Effect": "Allow",
     "Action": "sts:AssumeRole",
     "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
   }
   ```

   Ganti ARN dalam pernyataan dengan ARN dari peran yang dapat diasumsikan oleh pengguna.
6. Ikuti prompt pada layar untuk menyelesaikan pengubahan kebijakan.

Memperbarui kebijakan kepercayaan peran (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengubah siapa yang dapat mengambil peran.

Untuk mengubah kebijakan kepercayaan peran (AWS CLI)

1. (Opsional) Jika Anda tidak tahu nama peran yang ingin Anda ubah, jalankan perintah berikut untuk membuat daftar peran di akun Anda:
   - peran-daftar aws iam
2. (Opsional) Untuk melihat penjelasan peran terkini, gunakan perintah berikut:
   - aws sudah mendapatkan peran
3. Untuk mengubah prinsipal terpercaya yang dapat mengakses peran, buat file teks dengan kebijakan kepercayaan yang diperbarui. Anda dapat menggunakan editor teks apa pun untuk menyusun kebijakan.

   Misalnya, kebijakan kepercayaan berikut menunjukkan cara mereferensikan dua Akun AWS dalam `Principal` elemen. Ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk

mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, jalankan perintah berikut:

   - aws iam update-asumme-rolle-policy

Untuk mengizinkan pengguna di akun eksternal tepercaya menggunakan peran (AWS CLI)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Buat file JSON yang berisi kebijakan izin yang memberikan izin untuk mengasumsikan peran tersebut. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

   JSON

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

   Ganti ARN dalam pernyataan dengan ARN dari peran yang dapat diasumsikan oleh pengguna.

2. Jalankan perintah berikut untuk mengunggah file JSON yang berisi kebijakan kepercayaan ke IAM:

   - aws iam buat kebijakan

   Keluaran dari perintah ini mencakup ARN kebijakan. Catat ARN ini karena Anda akan memerlukannya di langkah berikutnya.

3. Tentukan pengguna atau kelompok mana yang akan dilampiri kebijakan ini. Jika Anda tidak tahu nama pengguna atau kelompok yang dimaksudkan, gunakan salah satu perintah berikut untuk membuat daftar pengguna atau kelompok di akun Anda:

   - pengguna daftar aws iam
   - grup daftar aws iam

4. Gunakan salah satu perintah berikut untuk melampirkan kebijakan yang Anda buat pada langkah sebelumnya kepada pengguna atau kelompok:

   - kebijakan lampiran pengguna aws iam
   - kebijakan grup lampiran aws iam

Memperbarui kebijakan kepercayaan peran (AWS API)

Anda dapat menggunakan AWS API untuk mengubah siapa yang dapat mengambil peran.

Untuk mengubah kebijakan kepercayaan peran (AWS API)

1. (Opsional) Jika Anda tidak tahu nama peran yang ingin Anda ubah, jalankan perintah berikut untuk membuat daftar peran di akun Anda:

   - ListRoles

2. (Opsional) Untuk melihat kebijakan kepercayaan terkini untuk peran, panggil operasi berikut:

   - GetRole

3. Untuk mengubah prinsipal terpercaya yang dapat mengakses peran, buat file teks dengan kebijakan kepercayaan yang diperbarui. Anda dapat menggunakan editor teks apa pun untuk menyusun kebijakan.

   Misalnya, kebijakan kepercayaan berikut menunjukkan cara mereferensikan dua Akun AWS dalam `Principal` elemen. Ini memungkinkan pengguna dalam dua terpisah Akun AWS untuk mengambil peran ini.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
```

```
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Jika Anda menetapkan prinsipal di akun lain, menambahkan akun ke kebijakan kepercayaan dari suatu peran hanya setengah dari membangun hubungan kepercayaan lintas akun. Secara default, tidak ada pengguna dalam akun terpercaya yang dapat mengasumsikan peran tersebut. Administrator untuk akun yang baru dipercaya harus memberikan izin kepada pengguna untuk mengasumsikan peran tersebut. Untuk melakukannya, administrator harus membuat atau mengubah kebijakan yang dilampirkan pada pengguna untuk memberikan akses kepada pengguna ke tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat prosedur berikut atau Berikan izin pengguna untuk beralih peran.

4. Untuk menggunakan file yang baru saja Anda buat untuk memperbarui kebijakan kepercayaan, panggil operasi berikut:

   - UpdateAssumeRolePolicy

Untuk mengizinkan pengguna di akun eksternal tepercaya menggunakan peran (AWS API)

Untuk informasi dan detail selengkapnya tentang prosedur ini, lihat Berikan izin pengguna untuk beralih peran.

1. Buat file JSON yang berisi kebijakan izin yang memberikan izin untuk mengasumsikan peran tersebut. Misalnya, kebijakan berikut memuat izin minimum yang diperlukan:

   JSON

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

```
    }
}
```

Ganti ARN dalam pernyataan dengan ARN dari peran yang dapat diasumsikan oleh pengguna.

2. Panggil operasi berikut untuk mengunggah file JSON yang berisi kebijakan kepercayaan ke IAM:

   - CreatePolicy

   Keluaran operasi ini mencakup ARN kebijakan. Catat ARN ini karena Anda akan memerlukannya di langkah berikutnya.

3. Tentukan pengguna atau kelompok mana yang akan dilampiri kebijakan ini. Jika Anda tidak tahu nama pengguna atau grup yang dimaksudkan, panggil salah satu operasi berikut untuk membuat daftar pengguna atau kelompok di akun Anda:

   - ListUsers
   - ListGroups

4. Panggil salah satu perintah berikut untuk melampirkan kebijakan yang Anda buat pada langkah sebelumnya kepada pengguna atau kelompok:

   - API: AttachUserPolicy
   - AttachGroupPolicy

Memperbarui izin untuk peran

Gunakan prosedur berikut untuk memperbarui kebijakan izin peran dan batasan izin.

Prasyarat: Lihat akses peran

Sebelum mengubah izin pengguna, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Memperbarui kebijakan izin untuk peran

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan kepercayaan untuk peran yang terkait dengan layanan dalam IAM. Anda mungkin dapat mengubah kebijakan izin dalam layanan

yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Memperbarui kebijakan izin peran (konsol)

Untuk mengubah izin yang diperbolehkan oleh peran (konsol)

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran.
3. Pilih nama peran yang ingin Anda ubah, dan kemudian pilih tabIzin.
4. Lakukan salah satu langkah berikut:

   - Untuk mengubah kebijakan yang dikelola oleh pelanggan yang sudah ada, pilih nama kebijakan tersebut dan kemudian pilih Ubah kebijakan.

     > **Note**
     >
     > Anda tidak dapat mengedit kebijakan AWS terkelola.
     > AWS kebijakan terkelola muncul dengan AWS ikon
     > ( ).
     > Untuk informasi selengkapnya tentang perbedaan antara kebijakan AWS terkelola dan kebijakan yang dikelola pelanggan, lihatKebijakan terkelola dan kebijakan inline.

   - Untuk melampirkan kebijakan terkelola yang ada ke peran, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.
   - Untuk mengedit kebijakan inline yang ada, perluas kebijakan dan pilih Edit.
   - Untuk menyematkan kebijakan sebaris baru, pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.
   - Untuk menghapus kebijakan yang ada dari peran, pilih kotak centang di samping nama kebijakan, lalu pilih Hapus.

Memperbarui kebijakan izin peran ()AWS CLI

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan kepercayaan untuk peran yang

terkait dengan layanan dalam IAM. Anda mungkin dapat mengubah kebijakan izin dalam layanan yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Untuk mengubah izin yang diperbolehkan oleh peran (AWS CLI)

1. (Opsional) Untuk melihat izin yang saat ini terkait dengan peran, jalankan perintah berikut:
    1. aws iam list-role-policies untuk mencantumkan kebijakan sebaris
    2. aws iam list-attached-role-policies untuk membuat daftar kebijakan terkelola
2. Perintah untuk memperbarui izin untuk peran berbeda-beda tergantung pada apakah Anda memperbarui kebijakan terkelola atau kebijakan inline.

    Untuk memperbarui kebijakan terkelola, panggil operasi berikut untuk membuat versi baru kebijakan terkelola:

    - aws iam create-policy-version

    Untuk memperbarui kebijakan inline, jalankan perintah berikut:

    - aws iam put-role-policy

Memperbarui kebijakan izin peran (AWS API)

Untuk mengubah izin yang diperbolehkan oleh peran tersebut, ubah kebijakan (atau beberapa kebijakan) izin peran tersebut. Anda tidak dapat mengubah kebijakan kepercayaan untuk peran yang terkait dengan layanan dalam IAM. Anda mungkin dapat mengubah kebijakan izin dalam layanan yang bergantung pada peran tersebut. Untuk memeriksa apakah layanan mendukung fitur ini, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya dalam kolom Peran yang terkait dengan layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut.

Untuk mengubah izin yang diizinkan oleh peran (AWS API)

1. (Opsional) Untuk melihat izin yang saat ini terkait dengan peran, panggil perintah berikut:
    1. ListRolePolicies untuk membuat daftar kebijakan inline

   2. ListAttachedRolePolicies untuk membuat daftar kebijakan terkelola
2. Operasi untuk memperbarui izin untuk peran berbeda-beda tergantung pada apakah Anda memperbarui kebijakan terkelola atau kebijakan inline.

   Untuk memperbarui kebijakan terkelola, panggil operasi berikut untuk membuat versi baru kebijakan terkelola:

   - CreatePolicyVersion

   Untuk memperbarui kebijakan inline, panggil operasi berikut:

   - PutRolePolicy

Perbarui batas izin untuk peran

Untuk mengubah izin maksimum yang diperbolehkan untuk peran, ubah batas izin peran.

Memperbarui batas izin peran (konsol)

Untuk mengubah kebijakan yang digunakan untuk mengatur batas izin untuk peran

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Pilih nama peran dengan batas izin yang ingin Anda ubah.
4. Pilih tab Izin. Jika perlu, buka bagian Batas izin lalu pilih Ubah batas.
5. Pilih kebijakan yang ingin Anda gunakan untuk batasan izin.
6. Pilih Ubah batas.

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini.

Memperbarui batas izin peran ()AWS CLI

Untuk mengubah kebijakan terkelola yang digunakan untuk mengatur batas izin peran (AWS CLI)

1. (Opsional) Untuk melihat batas izin terkini untuk peran, jalankan perintah berikut:

   - aws iam mendapatkan peran

2. Untuk menggunakan kebijakan terkelola yang berbeda untuk memperbarui batas izin untuk peran, jalankan perintah berikut:

   - aws iam put-role-permissions-boundary

   Peran hanya dapat memiliki satu kebijakan terkelola yang diatur sebagai batas izin. Jika Anda mengubah batas izin, Anda juga mengubah izin maksimum yang diperbolehkan untuk peran.

Memperbarui batas izin peran (API)AWS

Untuk mengubah kebijakan terkelola yang digunakan untuk menyetel batas izin untuk peran (API)AWS

1. (Opsional) Untuk melihat batas izin terkini untuk peran, panggil operasi berikut:

   - GetRole

2. Untuk menggunakan kebijakan terkelola untuk memperbarui batas izin untuk peran, panggil operasi berikut:

   - PutRolePermissionsBoundary

   Peran hanya dapat memiliki satu kebijakan terkelola yang diatur sebagai batas izin. Jika Anda mengubah batas izin, Anda juga mengubah izin maksimum yang diperbolehkan untuk peran.

Perbarui setelan untuk peran

Gunakan prosedur berikut untuk memperbarui deskripsi peran atau mengubah durasi sesi maksimum untuk peran.

Memperbarui deskripsi peran

Untuk mengubah deskripsi peran, ubah teks deskripsi.

Memperbarui deskripsi peran (konsol)

Untuk mengubah deskripsi peran (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi konsol IAM, pilih Peran.
3. Pilih nama peran yang akan diubah.
4. Di bagian Ringkasan, pilih Edit.
5. Memasukkan Deskripsi baru di kotak, dan memilih Simpan perubahan.

Memperbarui deskripsi peran (AWS CLI)

Untuk mengubah deskripsi peran (AWS CLI)

1. (Opsional) Untuk melihat deskripsi peran terkini, jalankan perintah berikut:
   - aws iam mendapatkan peran
2. Untuk memperbarui deskripsi peran, jalankan perintah berikut dengan parameter deskripsi:
   - peran pembaruan aws iam

Memperbarui deskripsi peran (AWS API)

Untuk mengubah deskripsi peran (AWS API)

1. (Opsional) Untuk melihat desripsi terkini untuk peran, panggil operasi berikut:
   - GetRole
2. Untuk memperbarui deskripsi peran, panggil operasi berikut dengan parameter deskripsi:
   - UpdateRole

Memperbarui durasi sesi maksimum untuk peran

Untuk menentukan setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan konsol, API AWS CLI, atau AWS API, ubah nilai setelan durasi sesi maksimum. Pengaturan ini dapat memiliki nilai dari 1 jam hingga 12 jam. Jika Anda tidak menentukan nilai, defaultnya maksimum 1 jam diterapkan. Pengaturan ini tidak membatasi sesi yang diasumsikan oleh AWS layanan.

Memperbarui durasi sesi peran maksimum (konsol)

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan konsol, AWS CLI, atau AWS API (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran.
3. Pilih nama peran yang akan diubah.
4. Di bagian Ringkasan, pilih Edit.
5. Untuk Durasi sesi maksimum, pilih nilai. Atau, pilih Durasi kustom dan masukkan nilai (dalam detik).
6. Pilih Simpan perubahan.

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi keamanan sementara peran IAM.

Dalam Konsol Manajemen AWS, sesi pengguna IAM adalah 12 jam secara default. Pengguna IAM yang beralih peran di konsol diberikan durasi sesi maksimum peran, atau sisa waktu dalam sesi pengguna, mana yang lebih sedikit.

Siapa pun yang mengambil peran dari AWS CLI atau AWS API dapat meminta sesi yang lebih lama, hingga maksimum ini. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta.

- Untuk menentukan durasi sesi menggunakan AWS CLI menggunakan `duration-seconds` parameter. Untuk mempelajari selengkapnya, lihat Beralih ke peran IAM (AWS CLI).
- Untuk menentukan durasi sesi menggunakan AWS API, gunakan `DurationSeconds` parameter. Untuk mempelajari selengkapnya, lihat Beralih ke peran IAM (AWS API).

Memperbarui durasi sesi peran maksimum (AWS CLI)

> **Note**
>
> Siapa pun yang mengambil peran dari AWS CLI atau API dapat menggunakan parameter `duration-seconds` CLI atau parameter API untuk meminta sesi `DurationSeconds`

yang lebih lama. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta dengan menggunakan parameter `DurationSeconds`. Jika pengguna tidak menentukan nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan mereka berlaku selama satu jam.

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan AWS CLI (AWS CLI)

1. (Opsional) Untuk melihat durasi sesi maksimum peran untuk peran, jalankan perintah berikut:

   - aws iam mendapatkan peran

2. Untuk memperbarui pengaturan durasi sesi maksimum peran, jalankan perintah berikut dengan parameter `max-session-duration` CLI atau parameter API `MaxSessionDuration`:

   - peran pembaruan aws iam

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi keamanan sementara peran IAM.

Memperbarui durasi sesi peran maksimum (AWS API)

> **Note**
>
> Siapa pun yang mengambil peran dari AWS CLI atau API dapat menggunakan parameter `duration-seconds` CLI atau parameter API untuk meminta sesi `DurationSeconds` yang lebih lama. Pengaturan `MaxSessionDuration` menentukan durasi maksimum dari sesi peran yang dapat diminta dengan menggunakan parameter `DurationSeconds`. Jika pengguna tidak menentukan nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan mereka berlaku selama satu jam.

Untuk mengubah setelan durasi sesi maksimum untuk peran yang diasumsikan menggunakan API (AWS API)

1. (Opsional) Untuk melihat durasi sesi maksimum terkini untuk peran, panggil operasi berikut:

- GetRole

2. Untuk memperbarui pengaturan durasi sesi maksimum peran, panggil operasi berikut dengan `max-session-duration` Parameter CLI atau `MaxSessionDuration` Parameter API:

   - UpdateRole

   Perubahan Anda tidak berlaku sampai saat berikutnya seseorang mengasumsikan peran ini. Untuk mempelajari cara mencabut sesi yang ada untuk peran ini, lihat Mencabut kredensi keamanan sementara peran IAM.

Hapus peran atau profil contoh

Jika Anda tidak lagi memerlukan peran, kami sarankan Anda untuk menghapus peran tersebut dan izin terkaitnya. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif.

Jika peran tersebut dikaitkan dengan instans EC2, Anda juga dapat menghapus peran tersebut dari profil instans dan kemudian menghapus profil instans.

> **Warning**
>
> Pastikan Anda tidak memiliki instans Amazon EC2 yang sedang berjalan dengan peran atau profil instans yang akan Anda hapus. Menghapus peran atau profil instans yang terkait dengan suatu instans yang sedang berjalan akan merusak aplikasi yang sedang berjalan pada instans tersebut.

Jika Anda memilih untuk tidak menghapus peran secara permanen, Anda dapat menonaktifkan peran. Untuk melakukannya, ubah kebijakan peran lalu cabut semua sesi saat ini. Misalnya, Anda dapat menambahkan kebijakan ke peran yang menolak akses ke semua peran AWS. Anda juga dapat mengubah kebijakan kepercayaan untuk menolak akses ke siapa pun yang mencoba untuk mengasumsikan peran tersebut. Untuk informasi lebih lanjut tentang mencabut sesi, lihat Mencabut kredensi keamanan sementara peran IAM.

Topik

- Melihat akses peran
- Menghapus peran yang terkait dengan layanan

- Menghapus peran IAM (konsol)
- Menghapus peran IAM ()AWS CLI
- Menghapus peran IAM (API)AWS
- Informasi Terkait

Melihat akses peran

Sebelum menghapus peran, kami menyarankan agar Anda meninjau kapan peran tersebut terakhir kali digunakan. Anda dapat melakukan ini menggunakan Konsol Manajemen AWS, the AWS CLI, atau AWS API. Anda harus melihat informasi ini karena Anda tidak ingin menghapus akses dari seseorang yang menggunakan peran tersebut.

Tanggal aktivitas terakhir peran mungkin tidak cocok dengan tanggal terakhir yang dilaporkan di tab Terakhir Diakses. Tab Terakhir Diakses melaporkan aktivitas hanya untuk layanan yang diizinkan oleh kebijakan izin peran. Tanggal aktivitas terakhir peran mencakup upaya terakhir untuk mengakses layanan apa pun AWS.

> **Note**
>
> Periode pelacakan untuk aktivitas terakhir peran dan data Terakhir Diakses adalah selama 400 hari terakhir. Periode ini bisa lebih singkat jika Wilayah Anda mulai mendukung fitur-fitur ini tahun lalu. Peran ini mungkin telah digunakan lebih dari 400 hari yang lalu. Untuk informasi lebih lanjut tentang periode pelacakan ini, lihat Tempat AWS melacak informasi terakhir yang diakses.

Untuk melihat kapan peran terakhir digunakan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Cari baris peran dengan aktivitas ingin Anda lihat. Anda dapat menggunakan bidang pencarian untuk mempersempit hasil. Lihat kolom Aktivitas terakhir untuk melihat jumlah hari sejak peran terakhir digunakan. Jika peran tersebut belum digunakan dalam periode pelacakan, maka tabel akan menampilkan Tidak ada.
4. Pilih nama peran untuk melihat informasi lebih lanjut. Halaman Ringkasan peran juga mencakup Aktivitas terakhir, yang menampilkan tanggal terakhir peran digunakan. Jika peran tersebut tidak

digunakan dalam 400 hari terakhir, maka Aktivitas terakhir menampilkan Tidak diakses dalam periode pelacakan.

Untuk menampilkan waktu terakhir peran digunakan (AWS CLI)

`aws iam get-role` - Jalankan perintah ini untuk mengembalikan informasi tentang peran, termasuk objek`RoleLastUsed`. Objek ini berisi `LastUsedDate` dan `Region` yang perannya terakhir digunakan. Jika `RoleLastUsed` ada, tetapi tidak berisi nilai, maka peran tersebut belum pernah digunakan dalam periode pelacakan.

Untuk melihat kapan peran terakhir digunakan (AWS API)

`GetRole` - Panggil operasi ini untuk mengembalikan informasi tentang peran, termasuk objek`RoleLastUsed`. Objek ini berisi `LastUsedDate` dan `Region` yang perannya terakhir digunakan. Jika `RoleLastUsed` ada, tetapi tidak berisi nilai, maka peran tersebut belum pernah digunakan dalam periode pelacakan.

Menghapus peran yang terkait dengan layanan

Metode yang Anda gunakan untuk menghapus peran terkait layanan bergantung pada layanan. Dalam beberapa kasus, Anda tidak perlu menghapus peran terkait layanan secara manual. Misalnya, ketika Anda menyelesaikan tindakan tertentu (seperti menghapus sumber daya) dalam layanan, layanan mungkin akan menghapus peran terkait layanan untuk Anda. Dalam kasus lain, layanan mungkin mendukung penghapusan peran terkait layanan secara manual dari konsol layanan, API, atau AWS CLI.

Tinjau dokumentasi untuk peran terkait layanan di layanan tertaut untuk mempelajari cara menghapus peran. Anda dapat melihat peran yang terkait dengan layanan di akun Anda dengan masuk ke Peran IAM di konsol. Peran yang terkait dengan layanan muncul dengan (Peran yang terkait dengan layanan) dalam kolom Entitas terpercaya di tabel. Banner pada halaman Ringkasan peran juga menunjukkan bahwa peran tersebut adalah peran yang terkait dengan layanan.

Jika layanan tidak menyertakan dokumentasi untuk menghapus peran terkait layanan, Anda dapat menggunakan konsol IAM AWS CLI, atau API untuk menghapus peran tersebut.

Menghapus peran IAM (konsol)

Saat Anda menggunakan Konsol Manajemen AWS untuk menghapus peran, IAM secara otomatis melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Ini juga secara otomatis

menghapus kebijakan sebaris apa pun yang terkait dengan peran, dan profil instans Amazon EC2 apa pun yang berisi peran tersebut.

> **Important**
>
> Dalam beberapa kasus, peran mungkin berkaitan dengan profil instans Amazon EC2, dan peran serta profil instans mungkin memiliki nama yang sama. Dalam hal ini Anda dapat menggunakan Konsol Manajemen AWS untuk menghapus peran dan profil instance. Pertautan ini terjadi secara otomatis untuk peran dan profil instans yang Anda buat di konsol. Jika Anda membuat peran dari AWS CLI, Alat untuk Windows PowerShell, atau AWS API, maka peran dan profil instance mungkin memiliki nama yang berbeda. Jika demikian, Anda tidak dapat menggunakan konsol untuk menghapusnya. Sebagai gantinya, Anda harus menggunakan AWS CLI, Tools for Windows PowerShell, atau AWS API untuk terlebih dahulu menghapus peran dari profil instance. Kemudian, Anda harus melakukan langkah terpisah untuk menghapus peran tersebut.

Untuk menghapus peran (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran, lalu centang kotak di samping nama peran yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Di kotak dialog konfirmasi, tinjau informasi yang terakhir diakses, yang menunjukkan kapan masing-masing peran yang dipilih terakhir mengakses AWS layanan. Ini membantu Anda mengonfirmasi jika peran tersebut saat ini aktif. Jika Anda ingin melanjutkan, masukkan nama peran di kolom input teks dan pilih Hapus. Jika Anda yakin, Anda dapat melanjutkan dengan penghapusan meskipun informasi yang terakhir diakses masih dimuat.

> **Note**
>
> Anda tidak dapat menggunakan konsol untuk menghapus profil instans kecuali itu memiliki nama yang sama dengan peran. Profil instans dihapus sebagai bagian dari proses penghapusan peran sebagaimana dijelaskan dalam prosedur sebelumnya. Untuk menghapus profil instance tanpa juga menghapus peran, Anda harus menggunakan AWS CLI atau AWS API. Untuk informasi selengkapnya, silakan lihat bagian-bagian berikut ini.

Menghapus peran IAM ()AWS CLI

Saat Anda menggunakan AWS CLI untuk menghapus peran, Anda harus terlebih dahulu menghapus kebijakan sebaris yang terkait dengan peran tersebut. Anda juga harus melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Jika Anda ingin menghapus profil instance terkait yang berisi peran, Anda harus menghapusnya secara terpisah.

Untuk menghapus peran (AWS CLI)

1. Jika Anda tidak tahu nama peran yang ingin Anda jalankan, jalankan perintah berikut untuk membuat daftar peran di akun Anda:

   ```
   aws iam list-roles
   ```

   Daftar tersebut mencakup Amazon Resource Name (ARN) dari setiap peran. Gunakan nama peran, bukan ARN, untuk merujuk ke peran dengan perintah CLI. Misalnya, jika peran memiliki ARN berikut: `arn:aws:iam::123456789012:role/myrole`, referensi Anda ke peran sebagai **`myrole`**.
2. Hapus peran dari semua profil instance yang terkait dengan peran tersebut.

   a. Untuk membuat daftar semua profil instans berkaitan dengan peran tersebut, masukkan perintah berikut:

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. Untuk menghapus peran dari profil instans, masukkan perintah berikut untuk setiap profil instans:

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```
3. Hapus semua kebijakan yang terkait dengan peran tersebut.

   a. Untuk mencantumkan semua kebijakan inline yang ada dalam peran, masukkan perintah berikut:

      ```
      aws iam list-role-policies --role-name role-name
      ```

   b. Untuk menghapus setiap kebijakan sebaris dari peran, masukkan perintah berikut untuk setiap kebijakan:

```
aws iam delete-role-policy --role-name role-name --policy-name policy-name
```

c. Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke peran, masukkan perintah berikut:

```
aws iam list-attached-role-policies --role-name role-name
```

d. Untuk melepaskan setiap kebijakan terkelola dari peran, masukkan perintah berikut untuk setiap kebijakan:

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. Masukkan perintah berikut untuk menghapus peran:

```
aws iam delete-role --role-name role-name
```

5. Jika Anda tidak berencana untuk menggunakan ulang profil instans yang terkait dengan peran tersebut, Anda dapat memasukkan perintah berikut untuk menghapusnya:

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Menghapus peran IAM (API)AWS

Saat Anda menggunakan API IAM untuk menghapus peran, Anda harus terlebih dahulu menghapus kebijakan sebaris yang terkait dengan peran tersebut. Anda juga harus melepaskan kebijakan terkelola yang terkait dengan peran tersebut. Jika Anda ingin menghapus profil instance terkait yang berisi peran, Anda harus menghapusnya secara terpisah.

Untuk menghapus peran (AWS API)

1. Untuk mencantumkan semua profil instance yang terkait dengan peran, hubungi ListInstanceProfilesForRole.

   Untuk menghapus peran dari profil instance, panggil RemoveRoleFromInstanceProfile. Anda harus meneruskan nama peran dan nama profil instans.

   Jika Anda tidak akan menggunakan ulang profil instans yang terkait dengan peran tersebut, panggil DeleteInstanceProfile untuk menghapusnya.

2. Untuk mencantumkan semua kebijakan sebaris untuk peran, hubungi ListRolePolicies.

   Untuk menghapus kebijakan sebaris yang terkait dengan peran, panggil DeleteRolePolicy. Anda harus meneruskan nama peran dan nama kebijakan sebaris.

3. Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke peran, hubungi ListAttachedRolePolicies.

   Untuk melepaskan kebijakan terkelola yang melekat pada peran, hubungi DetachRolePolicy. Anda harus melewati nama peran dan kebijakan terkelola ARN.

4. Panggil DeleteRole untuk menghapus peran.

Informasi Terkait

Untuk informasi umum tentang profil instans, lihat Gunakan profil instance.

Untuk informasi umum tentang peran yang terkait dengan layanan, lihat Buat peran tertaut layanan.

Metode untuk mengambil peran

Sebelum pengguna, aplikasi, atau layanan dapat menggunakan peran yang Anda buat, Anda harus memberikan izin untuk beralih ke peran tersebut. Anda dapat menggunakan kebijakan apa pun yang dilampirkan ke grup atau pengguna untuk memberikan izin yang diperlukan. Setelah izin diberikan, pengguna dapat mengambil peran dari Konsol Manajemen AWS, Alat untuk Windows PowerShell, AWS Command Line Interface (AWS CLI) dan `AssumeRole`API.

> **Important**
>
> Saat Anda membuat program peran secara terprogram, alih-alih di konsol IAM, Anda memiliki opsi untuk menambahkan `Path` hingga 512 karakter selain `RoleName`, yang panjangnya dapat mencapai 64 karakter. Namun, jika Anda bermaksud menggunakan peran dengan fitur Switch Role di Konsol Manajemen AWS, maka gabungan `Path` dan `RoleName` tidak dapat melebihi 64 karakter.

Metode yang digunakan untuk mengambil peran menentukan siapa yang dapat mengambil peran dan berapa lama sesi peran dapat berlangsung. Saat menggunakan operasi `AssumeRole*` API, peran IAM yang Anda asumsikan adalah sumber daya. Pengguna atau peran yang memanggil operasi `AssumeRole*` API adalah prinsipnya.

Tabel berikut membandingkan metode untuk mengasumsikan peran.

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min | maks | default)
Konsol Manajemen AWS	Pengguna atau peran[1] (dengan beralih peran)	Durasi sesi maksimum pada Halaman ringkasan Peran	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
Operasi `assume-role` CLI atau `AssumeRole` API	Pengguna atau peran[1]	Parameter `duration-seconds` CLI atau `DurationSeconds` API	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
Operasi `assume-role-with-saml` CLI atau `AssumeRoleWithSAML` API	Pengguna yang diotentikasi menggunakan SAML	Parameter `duration-seconds` CLI atau `DurationSeconds` API	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
Operasi `assume-role-with-web-identity` CLI atau `AssumeRoleWithWebIdentity` API	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	Parameter `duration-seconds` CLI atau `DurationSeconds` API	15 menit | Pengaturan durasi sesi maksimum[2] | 1 jam
URL Konsol yang dibangun	Pengguna atau peran	`SessionDuration`	15 menit | 12 jam | 1 jam

Metode untuk mengasumsikan peran	Siapa yang bisa mengambil peran	Metode untuk menentukan masa pakai kredensi	Masa pakai kredensi (min | maks | default)
dengan `AssumeRole`		Parameter HTML dalam URL	
URL Konsol yang dibangun dengan `AssumeRoleWithSAML`	Pengguna yang diotentikasi menggunakan SAML	`SessionDuration` Parameter HTML dalam URL	15 menit | 12 jam | 1 jam
URL Konsol yang dibangun dengan `AssumeRoleWithWebIdentity`	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	`SessionDuration` Parameter HTML dalam URL	15 menit | 12 jam | 1 jam

[1] Menggunakan kredensi dari satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, durasi sesi peran dibatasi hingga satu jam. Ini berlaku untuk peralihan Konsol Manajemen AWS peran AWS CLI, dan operasi API. Batasan ini tidak berlaku untuk asumsi awal peran dari kredenal pengguna, atau aplikasi yang berjalan di EC2 instans Amazon yang menggunakan profil instans.

[2] Pengaturan ini dapat memiliki nilai dari 1 jam hingga 12 jam. Untuk detail tentang pengubahan pengaturan durasi sesi maksimum, lihat Manajemen peran IAM. Setelan ini menentukan durasi sesi maksimum yang dapat Anda minta saat Anda mendapatkan kredensial peran. Misalnya, saat Anda menggunakan AssumeRole* Operasi API untuk mengasumsikan peran, Anda dapat menentukan durasi sesi dengan menggunakan parameter `DurationSeconds`. Gunakan parameter ini untuk menentukan panjang durasi sesi peran mulai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Pengguna IAM yang beralih peran di konsol diberikan durasi sesi maksimum, atau sisa waktu dalam sesi pengguna mereka, mana yang lebih sedikit. Anggaplah Anda menetapkan durasi maksimum 5 jam dalam suatu peran. Pengguna IAM yang telah masuk ke konsol selama 10 jam (dari nilai maksimum standar 12) berganti peran. Durasi sesi peran yang tersedia

adalah 2 jam. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran nanti di halaman ini.

> **Catatan**
>
> - Pengaturan durasi sesi maksimum tidak membatasi sesi yang diambil oleh layanan AWS .
> - Kredensi peran Amazon EC2 IAM tidak tunduk pada durasi sesi maksimum yang dikonfigurasi dalam peran.
> - Untuk memungkinkan pengguna untuk mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti Amazon EC2, Amazon ECS, Amazon EKS, dan Lambda memberikan kredensi sementara dan secara otomatis memperbarui kredensional ini. Ini memastikan bahwa Anda selalu memiliki seperangkat kredensional yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredensi sementara. Namun, jika Anda berniat untuk meneruskan tag sesi atau kebijakan sesi, Anda perlu mengambil peran saat ini lagi. Untuk mempelajari cara memodifikasi kebijakan kepercayaan peran untuk menambahkan peran utama ARN atau Akun AWS ARN, lihat. Memperbarui kebijakan kepercayaan peran

Topik

- Beralih dari pengguna ke peran IAM (konsol)
- Beralih ke peran IAM (AWS CLI)
- Beralih ke peran IAM (Alat untuk Windows PowerShell)
- Beralih ke peran IAM (AWS API)
- Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2
- Gunakan profil instance

Beralih dari pengguna ke peran IAM (konsol)

Anda dapat beralih peran saat masuk sebagai pengguna IAM, pengguna di Pusat Identitas IAM, SAML-federated peran, atau peran federasi identitas web. Per an menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses AWS sumber daya yang Anda butuhkan. Namun, Anda tidak masuk ke peran, tetapi setelah masuk sebagai pengguna IAM, Anda dapat beralih ke peran

IAM. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat Peran IAM, dan Pembuatan peran IAM.

Izin pengguna Anda dan peran apa pun yang Anda alihkan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda beralih ke suatu peran, sementara Anda meninggalkan izin pengguna dan bekerja dengan izin yang diberikan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Saat Anda mengganti peran di Konsol Manajemen AWS, konsol selalu menggunakan kredenSIAL asli Anda untuk mengotorisasi sakelar. Misalnya, jika Anda beralih ke rolEA, IAM menggunakan kredensi asli Anda untuk menentukan apakah Anda diizinkan untuk mengasumsikan rolEA. Jika Anda kemudian beralih ke RoleB saat Anda menggunakan rolEA, AWS masih menggunakan kredenSIAL asli Anda untuk mengotorisasi sakelar, bukan kredenSIAL untuk rolEA.

> **Note**
>
> Saat Anda masuk sebagai pengguna di Pusat Identitas IAM, sebagai SAML-federated peran, atau sebagai peran federasi identitas web, Anda mengambil peran IAM saat memulai sesi. Misalnya, ketika pengguna di Pusat Identitas IAM masuk ke portal AWS akses, mereka dapat memilih kumpulan izin atau mengakses peran yang ditetapkan melalui manajer akses akun sebelum mereka dapat mengakses AWS sumber daya.

Sesi peran

Saat Anda beralih peran, Konsol Manajemen AWS sesi Anda berlangsung selama 1 jam secara default. Sesi pengguna IAM adalah 12 jam secara default, pengguna lain mungkin memiliki durasi sesi yang berbeda yang ditentukan. Saat Anda mengganti peran di konsol, Anda diberikan durasi sesi maksimum peran, atau sisa waktu dalam sesi pengguna Anda, mana saja yang kurang. Anda tidak dapat memperpanjang durasi sesi dengan mengambil peran.

Misalnya, anggaplah durasi sesi maksimum 10 jam ditetapkan untuk peran. Anda telah masuk ke konsol selama 8 jam ketika Anda memutuskan untuk beralih ke peran. Ada 4 jam tersisa dalam sesi pengguna Anda, jadi durasi sesi peran yang diizinkan adalah 4 jam, bukan durasi sesi maksimum 10 jam. Tabel berikut menunjukkan cara menentukan durasi sesi untuk pengguna IAM saat berganti peran di dalam konsol.

Waktu sesi pengguna IAM yang tersisa adalah...	Durasi sesi peran adalah...		
Kurang dari durasi sesi maksimum peran	Waktu yang tersisa di sesi pengguna		
Lebih dari durasi sesi maksimum peran	Nilai durasi sesi maksimum		
Setara dengan durasi sesi maksimum peran	Nilai durasi sesi maksimum (perkiraan)		

Menggunakan kredenSIAL dari satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, durasi sesi dibatasi hingga satu jam, terlepas dari pengaturan durasi sesi maksimum yang dikonfigurasi untuk peran individual. Ini berlaku untuk Konsol Manajemen AWS pengalihan peran, AWS CLI, dan operasi API.

> **Note**
>
> Beberapa konsol AWS layanan dapat memperbarui ulang sesi peran Anda secara otomatis saat kedaluwarsa tanpa Anda mengambil tindakan apa pun. Beberapa mungkin akan meminta Anda memuat ulang halaman peramban untuk mengotentikasi ulang sesi Anda.

Pertimbangan-pertimbangan

- Anda tidak dapat beralih peran jika Anda masuk sebagai Pengguna root akun AWS.
- Pengguna harus diberi izin untuk beralih peran berdasarkan kebijakan. Untuk petunjuk, lihat Berikan izin pengguna untuk beralih peran.
- Anda tidak dapat mengganti peran Konsol Manajemen AWS ke peran yang membutuhkan ExternalId nilai. Anda hanya dapat beralih ke peran tersebut dengan memanggil `AssumeRole` API yang mendukung parameter `ExternalId`.

Untuk beralih ke peran

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan di Masuk ke Konsol Manajemen AWS dalam Panduan AWS Sign-In Pengguna.
2. Di Konsol Manajemen AWS, pilih nama pengguna Anda di bilah navigasi di kanan atas. Biasanya terlihat seperti ini: ***username*** @ ***account_ID_number_or_alias***.
3. Pilih salah satu metode berikut untuk beralih peran:
   - Pilih Beralih peran.
   - Jika Anda telah memilih untuk mendukung multi-sesi, pilih Tam bah sesi dan pilih Ber alih peran.

   > **Note**
   >
   > Anda dapat masuk hingga lima identitas berbeda secara bersamaan dalam satu browser web di Konsol Manajemen AWS. Untuk detailnya, lihat Masuk ke beberapa akun di Panduan Mem Konsol Manajemen AWS ulai.

4. Pada halaman Ganti Peran, ketik nomor ID akun atau alias akun dan nama peran yang diberikan oleh administrator Anda.

   > **Note**
   >
   > Jika administrator Anda membuat peran itu dengan sebuah jalur, seperti `division_abc/subdivision_efg/roleToDoX`, maka Anda harus mengetik jalur dan nama lengkap tersebut di kotak Peran. Jika Anda hanya mengetik nama peran, atau jika gabungan `Path` dan `RoleName` lebih dari 64 karakter, pergantian peran gagal. Ini adalah batasan cookie peramban yang menyimpan nama peran tersebut. Jika ini terjadi, hubungi administrator Anda dan minta mereka untuk mengurangi ukuran jalur dan nama peran.

5. (Opsional) Anda dapat memasukkan nama tampilan dan memilih warna tampilan yang akan menyorot peran di bilah navigasi konsol.
   - Untuk Nama tampilan, ketik teks yang ingin Anda tampilkan di bilah navigasi sebagai pengganti nama pengguna Anda saat peran ini aktif. Ada saran nama, berdasarkan informasi akun dan peran, tetapi Anda dapat mengubahnya menjadi apa pun yang berarti bagi Anda.
   - Untuk Warna tampilan, pilih warna untuk menyorot nama tampilan.

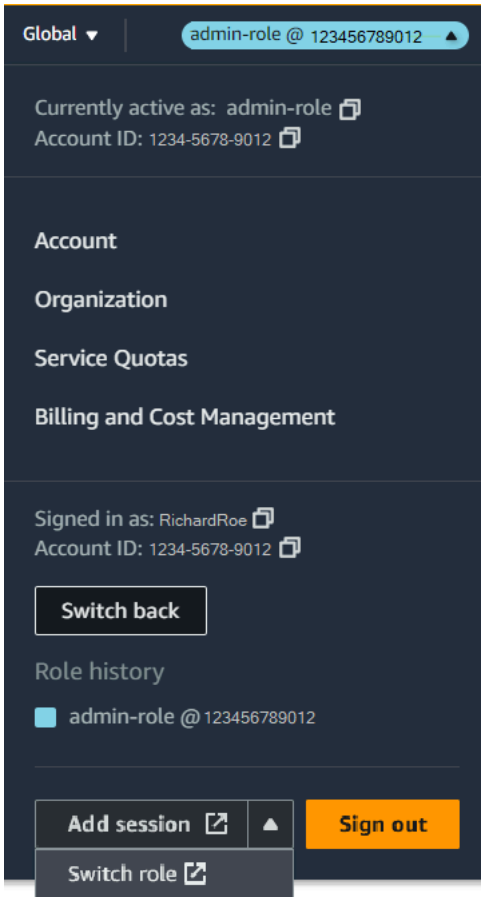
Nama dan warna dapat membantu mengingatkan Anda saat peran ini aktif, yang mengubah izin Anda. Misalnya, untuk peran yang memberi Anda akses ke lingkungan pengujian, Anda dapat menentukan Nama tampilan dari **`Test`** dan pilih Warna hijau. Untuk peran yang memberi Anda akses ke produksi, Anda dapat menentukan Nama tampilan dari **`Production`** dan pilih merah sebagai Warna.

6. Pilih Ganti Peran. Nama tampilan dan warna menggantikan nama pengguna Anda pada bilah navigasi, dan Anda dapat mulai menggunakan izin yang diberikan peran tersebut kepada Anda.
7. Setelah menyelesaikan tugas yang memerlukan peran IAM, Anda dapat beralih kembali ke sesi awal. Ini akan menghapus izin tambahan yang disediakan oleh peran dan mengembalikan Anda ke izin standar Anda.

   a. Di konsol IAM, pilih Nama tampilan peran Anda di bilah navigasi di kanan atas.

   b. Pilih Beralih kembali.

      Misalnya, anggap Anda masuk ke nomor akun `123456789012` menggunakan nama pengguna `Richard`. Setelah menggunakan peran `admin-role`, Anda ingin berhenti menggunakan peran dan kembali ke izin awal. Untuk berhenti menggunakan peran, pilih admin-role @ 1234567890 12, lalu pilih Beralih kembali.



> **Tip**
>
> Beberapa peran terakhir yang Anda gunakan muncul pada menu. Lain kali Anda ingin beralih ke salah satu peran tersebut, Anda cukup memilih peran yang Anda inginkan. Anda hanya diminta untuk mengetik akun dan informasi peran secara manual jika peran tidak ditampilkan di menu.

Sumber daya tambahan

- Berikan izin pengguna untuk beralih peran
- Memberikan izin pengguna untuk meneruskan peran ke AWS layanan
- Buat peran untuk memberikan izin kepada pengguna IAM
- Buat peran untuk mendelegasikan izin ke AWS layanan
- Memecahkan masalah peran IAM

Beralih ke peran IAM (AWS CLI)

Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS yang Anda perlukan. Dalam pengertian itu, ini mirip dengan pengguna di AWS Identity and Access Management (IAM). Saat Anda masuk sebagai pengguna, Anda mendapatkan serangkaian izin tertentu. Namun, Anda tidak masuk ke sebuah peran, tetapi setelah masuk sebagai pengguna, Anda dapat beralih ke sebuah peran. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat Peran IAM, dan Pembuatan peran IAM. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat Metode untuk mengambil peran.

> **Important**
>
> Izin pengguna IAM Anda dan peran apa pun yang Anda asumsikan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda mengasumsikan suatu peran, sementara waktu Anda meninggalkan izin pengguna dan bekerja dengan izin yang ditetapkan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Anda dapat menggunakan peran untuk menjalankan AWS CLI perintah saat Anda masuk sebagai pengguna IAM. Anda juga dapat menggunakan peran untuk menjalankan AWS CLI perintah saat Anda masuk sebagai pengguna yang diautentikasi secara eksternal (SAML atau OIDC) yang sudah menggunakan peran. Selain itu, Anda dapat menggunakan peran untuk menjalankan AWS CLI perintah dari dalam instans Amazon EC2 yang dilampirkan ke peran melalui profil instansnya. Anda tidak dapat mengambil peran saat masuk sebagai Pengguna root akun AWS.

Rantaian peran — Anda juga dapat menggunakan rantai peran, yang menggunakan izin dari peran untuk mengakses peran kedua.

Secara default, sesi peran Anda berlangsung selama satu jam. Saat Anda mengasumsikan peran dengan menggunakan `assume-role*` operasi CLI, Anda dapat menentukan nilai untuk parameter `duration-seconds`. Nilai ini dapat berkisar dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Jika Anda mengganti peran di konsol, durasi sesi Anda dibatasi maksimal satu jam. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Jika Anda kemudian menggunakan parameter `duration-seconds` untuk memberikan nilai lebih dari satu jam, operasi gagal.

Skenario contoh: Beralih ke peran produksi

Bayangkan bahwa Anda adalah pengguna IAM untuk bekerja di lingkungan pengembangan. Dalam skenario ini, Anda terkadang perlu bekerja dengan lingkungan produksi pada baris perintah dengan AWS CLI. Anda sudah memiliki kredensial access key yang tersedia untuk Anda. Ini dapat berupa key pair akses yang ditetapkan untuk pengguna IAM standar Anda. Atau, jika Anda masuk sebagai prinsipal federasi SAML atau OIDC, itu bisa menjadi pasangan kunci akses untuk peran yang awalnya ditugaskan kepada Anda. Jika izin Anda saat ini memberi Anda kemampuan untuk mengambil peran IAM tertentu, maka Anda dapat mengidentifikasi peran tersebut dalam "profil" dalam file AWS CLI konfigurasi. Perintah tersebut kemudian dijalankan atas izin peran IAM yang ditentukan, bukan identitas awal. Perhatikan bahwa ketika Anda menentukan profil itu dalam AWS CLI perintah, Anda menggunakan peran baru. Dalam situasi ini, Anda tidak dapat menggunakan izin awal dalam akun pengembangan pada saat bersamaan. Alasannya adalah bahwa hanya satu rangkaian izin yang dapat berlaku pada satu waktu.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan di AWS. Administrator Anda mungkin akan meminta Anda menentukan identitas sumber atau nama sesi peran ketika Anda mengambil peran tersebut. Untuk informasi selengkapnya, lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Untuk beralih ke peran produksi (AWS CLI)

1. Jika Anda belum pernah menggunakan AWS CLI, maka Anda harus terlebih dahulu mengonfigurasi profil CLI default Anda. Buka prompt perintah dan atur AWS CLI instalasi Anda untuk menggunakan kunci akses dari pengguna IAM Anda atau dari peran federasi Anda. Untuk informasi lebih lanjut, lihat Mengonfigurasi AWS Command Line Interface di Panduan Pengguna AWS Command Line Interface .

   Jalankan perintah aws configure sebagai berikut:

```
aws configure
```

Saat diminta, berikan informasi berikut:

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-east-2
Default output format [None]: json
```

2. Buat profil baru untuk peran dalam `.aws/config` file dalam Unix atau Linux, atau `C:\Users\USERNAME\.aws\config` file dalam Windows. Contoh berikut membuat sebuah profil bernama `prodaccess` yang beralih ke peran *`ProductionAccessRole`* dalam akun `123456789012`. Anda mendapatkan peran ARN dari administrator akun yang membuat peran tersebut. Ketika profil ini dipanggil, AWS CLI menggunakan kredensialnya `source_profile` untuk meminta kredenSIAL untuk peran tersebut. Oleh karena itu, identitas direferensikan karena `source_profile` harus memiliki izin `sts:AssumeRole` untuk peran yang ditentukan dalam `role_arn`.

```
[profile prodaccess]
    role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
    source_profile = default
```

3. Setelah Anda membuat profil baru, AWS CLI perintah apa pun yang menentukan parameter `--profile prodaccess` berjalan di bawah izin yang dilampirkan ke peran IAM al `ProductionAccessRole` ih-alih pengguna default.

```
aws iam list-users --profile prodaccess
```

Perintah ini berfungsi jika izin yang ditetapkan ke `ProductionAccessRole` mengaktifkan pencatuman pengguna di akun AWS saat ini.

4. Untuk kembali ke izin yang diberikan oleh kredensial asli Anda, jalankan perintah tanpa parameter `--profile`. Kem AWS CLI bali menggunakan kredensional di profil default Anda, yang Anda konfigurasikanStep 1.

Untuk informasi selengkapnya, lihat Mengambil Peran di Panduan AWS Command Line Interface Pengguna.

Skenario contoh: Menginzinkan peran profil instans untuk beralih ke peran dalam akun lain

Bayangkan Anda menggunakan dua Akun AWS, dan Anda ingin mengizinkan aplikasi yang berjalan pada instans Amazon EC2 untuk menjalankan AWS CLI perintah di kedua akun. Anggap bahwa instans EC2 ada di akun `111111111111`. Instans tersebut menyertakan peran profil `abcd` instance yang memungkinkan aplikasi melakukan tugas Amazon S3 yang hanya dapat dibaca pada `amzn-s3-demo-bucket1` bucket dalam akun yang sama`111111111111`. Namun, aplikasi tersebut juga harus diizinkan untuk mengambil peran lintas akun `efgh` untuk melakukan tugas di akun `222222222222`. Untuk melakukannya, peran profil instans EC2 `abcd` harus memiliki kebijakan izin berikut:

Kebijakan izin peran akun 1111111111 ***abcd***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
```

```
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Anggap bahwa peran lintas akun `efgh` memungkinkan tugas Amazon S3 hanya baca di bucket `amzn-s3-demo-bucket2` dalam akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Kebijakan izin peran akun 2222222222 ***efgh***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
```

```
        }
    ]
}
```

Peran `efgh` harus memungkinkan peran profil instans `abcd` untuk mengasumsikannya. Untuk melakukannya, peran `efgh` harus memiliki kebijakan kepercayaan berikut:

Kebijakan kepercayaan peran akun 2222222222 ***efgh***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Untuk kemudian menjalankan AWS CLI perintah di akun222222222222, Anda harus memperbarui file konfigurasi CLI. Identifikasi peran `efgh` sebagai "profil" dan peran profil instans EC2 `abcd` sebagai "sumber kredensial" dalam file konfigurasi AWS CLI . Kemudian perintah CLI Anda dijalankan dengan izin peran `efgh`, bukan peran `abcd` asli.

> **Note**
>
> Untuk tujuan keamanan, Anda dapat menggunakan AWS CloudTrail untuk mengaudit penggunaan peran dalam akun. Untuk membedakan sesi peran ketika peran digunakan oleh prinsipal yang berbeda di log CloudTrail , Anda dapat menggunakan nama sesi peran. Ketika mengambil AWS CLI peran atas nama pengguna seperti yang dijelaskan dalam topik ini, nama sesi peran secara otomatis dibuat sebagai`AWS-CLI-session-`*nnnnnnnn*. Berikut *nnnnnnnn* adalah bilangan bulat yang mewakili waktu dalam waktu zaman Unix (jumlah

detik sejak tengah malam UTC pada 1 Januari 1970). Untuk informasi selengkapnya, lihat CloudTrail Referensi Peristiwa di Panduan AWS CloudTrail Pengguna.

Untuk mengizinkan peran profil instans EC2 beralih ke peran lintas akun (AWS CLI)

1. Anda tidak perlu mengonfigurasi profil CLI default. Sebagai gantinya, Anda dapat memuat kredensial dari metadata profil instans EC2. Buat profil baru untuk peran dalam file `.aws/config`. Contoh berikut membuat profil `instancecrossaccount` yang beralih ke peran *`efgh`* dalam akun `222222222222`. Saat profil ini dipanggil, AWS CLI menggunakan kredensial metadata profil instans EC2 untuk meminta kredensial bagi peran tersebut. Karena itu, peran profil instans EC2 harus memiliki izin `sts:AssumeRole` ke peran yang ditentukan dalam `role_arn`.

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Setelah Anda membuat profil baru, AWS CLI perintah apa pun yang menentukan parameter `--profile instancecrossaccount` berjalan di bawah izin yang dilampirkan ke `efgh` peran dalam akun`222222222222`.

   ```
   aws s3 ls amzn-s3-demo-bucket2 --profile instancecrossaccount
   ```

   Perintah ini berfungsi jika izin yang ditetapkan ke `efgh` peran memungkinkan daftar pengguna saat ini Akun AWS.
3. Untuk kembali ke izin profil instans EC2 asli di akun `111111111111`, jalankan perintah CLI tanpa parameter `--profile`.

Untuk informasi selengkapnya, lihat Mengambil Peran di Panduan AWS Command Line Interface Pengguna.

Beralih ke peran IAM (Alat untuk Windows PowerShell)

Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS yang Anda perlukan. Dalam pengertian itu, ini mirip dengan pengguna di AWS Identity and Access Management (IAM). Saat Anda masuk sebagai pengguna, Anda mendapatkan serangkaian izin tertentu. Namun, Anda tidak masuk ke sebuah peran, tetapi setelah masuk, Anda dapat beralih

ke sebuah peran. Sementara waktu ini mengesampingkan izin pengguna awal Anda dan justru memberikan izin yang ditetapkan untuk peran. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat Peran IAM, dan Pembuatan peran IAM.

> **Important**
>
> Izin pengguna IAM Anda dan peran apa pun yang Anda asumsikan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda beralih ke suatu peran, sementara Anda meninggalkan izin pengguna dan bekerja dengan izin yang diberikan ke peran tersebut. Saat Anda keluar dari peran, izin pengguna asli Anda dipulihkan secara otomatis.

Bagian ini menjelaskan cara berganti peran ketika Anda bekerja di baris perintah dengan AWS Tools for Windows PowerShell.

Bayangkan Anda memiliki akun di lingkungan pengembangan dan kadang-kadang Anda perlu bekerja dengan lingkungan produksi di baris perintah menggunakan Tools untuk Windows PowerShell. Anda sudah memiliki satu set kredensial access key yang tersedia untuk Anda. Ini dapat berupa akses key pair yang ditetapkan untuk pengguna IAM standar Anda. Atau, jika Anda masuk sebagai prinsipal federasi SAML atau OIDC, mereka dapat menjadi pasangan kunci akses untuk peran yang awalnya ditugaskan kepada Anda. Anda dapat menggunakan kredensial ini untuk menjalankan `Use-STSRole` cmdlet yang melewati ARN peran baru sebagai parameter. Perintah tersebut mengembalikan kredensial keamanansementara untuk peran yang diminta. Kemudian, Anda dapat menggunakan kredensial tersebut dalam menurunkan perintah PowerShell dengan izin peran untuk mengakses sumber daya dalam produksi. Saat menggunakan peran tersebut, Anda tidak dapat menggunakan izin pengguna dalam akun Pengembangan karena hanya satu rangkaian izin yang berlaku pada satu waktu.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan di AWS. Administrator Anda mungkin akan meminta Anda menentukan identitas sumber atau nama sesi peran ketika Anda mengambil peran tersebut. Untuk informasi selengkapnya, lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Perhatikan bahwa semua kunci akses dan token hanyalah contoh dan tidak dapat digunakan seperti yang ditunjukkan. Ganti dengan nilai yang sesuai dari lingkungan langsung Anda.

Untuk beralih ke peran IAM (Alat untuk Windows PowerShell)

1. Buka prompt PowerShell perintah dan konfigurasikan profil default untuk menggunakan kunci akses dari pengguna IAM Anda saat ini atau dari peran federasi Anda. Jika sebelumnya Anda telah menggunakan Tools untuk Windows PowerShell, maka ini kemungkinan sudah dilakukan. Perhatikan bahwa Anda dapat beralih peran hanya jika Anda masuk sebagai pengguna IAM, bukan. Pengguna root akun AWS

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Untuk informasi selengkapnya, lihat Menggunakan AWS KredenSIAL di Panduan Alat AWS untuk PowerShell Pengguna.

2. Untuk mendapatkan kredensial peran baru, jalankan perintah berikut untuk beralih ke peran *RoleName* dalam akun 123456789012. Anda mendapatkan peran ARN dari administrator akun yang membuat peran tersebut. Perintah ini mengharuskan Anda untuk juga memberikan nama sesi. Anda dapat memilih teks apa pun untuk itu. Perintah berikut meminta kredensial dan kemudian menangkap objek properti `Credentials` dari objek hasil yang dikembalikan dan menyimpannya dalam variabel `$Creds`.

   ```
   PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
   RoleSessionName "MyRoleSessionName").Credentials
   ```

   `$Creds` adalah objek yang sekarang berisi elemen `AccessKeyId`, `SecretAccessKey`, dan `SessionToken` yang Anda butuhkan dalam langkah-langkah berikut ini. Perintah sampel berikut ini menggambarkan nilai-nilai umum:

   ```
   PS C:\> $Creds.AccessKeyId
   AKIAIOSFODNN7EXAMPLE

   PS C:\> $Creds.SecretAccessKey
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

   PS C:\> $Creds.SessionToken
   ```

```
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Untuk menggunakan kredensial ini untuk perintah berikutnya, sertakan mereka dengan parameter `-Credential`. Misalnya, perintah berikut menggunakan kredensial dari peran dan bekerja hanya jika peran tersebut diberik izin`iam:ListRoles` dan karenanya dapat menjalankan `Get-IAMRoles` cmdlet:

```
PS C:\> get-iamroles -Credential $Creds
```

4. Untuk kembali ke kredenSIAL asli Anda, cukup berhenti menggunakan `-Credentials $Creds` parameter dan izinkan PowerShell untuk kembali ke kredenSIAL yang disimpan di profil default.

Beralih ke peran IAM (AWS API)

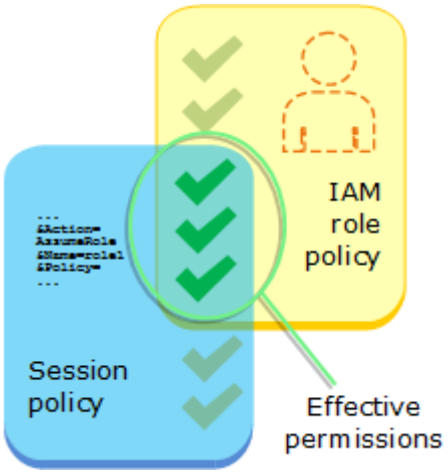
Peran menentukan serangkaian izin yang dapat Anda gunakan untuk mengakses sumber daya AWS . Dalam hal itu, ini mirip dengan Pengguna IAM. Seorang kepala sekolah (orang atau aplikasi) mengambil peran untuk menerima izin sementara untuk melaksanakan tugas yang diperlukan dan berinteraksi dengan AWS sumber daya. Perannya bisa di akun Anda sendiri atau lainnya Akun AWS. Untuk informasi lebih lanjut tentang peran, keuntungannya, dan cara membuatnya, lihat Peran IAM, dan Pembuatan peran IAM. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk mengasumsikan peran, lihat Metode untuk mengambil peran.

> **Important**
>
> Izin pengguna IAM Anda dan peran apa pun yang Anda asumsikan tidak bersifat kumulatif. Hanya satu rangkaian izin yang aktif pada satu waktu. Saat Anda mengasumsikan suatu peran, sementara waktu Anda meninggalkan izin pengguna dan bekerja dengan izin yang

ditetapkan ke peran tersebut. Saat Anda keluar dari peran tersebut, izin asli Anda akan dipulihkan secara otomatis.

Untuk mengambil peran, aplikasi memanggil operasi AWS STS `AssumeRole`API dan meneruskan ARN peran yang akan digunakan. Operasi membuat sesi baru dengan kredensial sementara. Sesi ini memiliki izin yang sama dengan kebijakan berbasis identitas untuk peran tersebut.

Saat Anda menelepon `AssumeRole`, Anda secara opsional dapat meneruskan kebijakan sesi inline atau terkelola. Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi kredensyal sementara secara terprogram untuk peran atau sesi pengguna gabungan. Anda dapat menyampaikan satu dokumen kebijakan sesi inline JSON menggunakan parameter `Policy`. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas entitas dan kebijakan sesi. Kebijakan sesi berguna saat Anda perlu memberikan kredensial sementara kepada orang lain. Anda dapat menggunakan kredensial sementara peran tersebut dalam panggilan API AWS berikutnya untuk mengakses sumber daya di akun yang memiliki peran tersebut. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas. Untuk mempelajari selengkapnya tentang cara AWS menentukan izin efektif suatu peran, lihatLogika evaluasi kebijakan.



Anda dapat memanggil `AssumeRole` saat Anda masuk sebagai pengguna IAM, atau sebagai pengguna terotentikasi secara eksternal (SAML atau OIDC) yang sudah menggunakan peran. Anda juga dapat menggunakan rantai peran, yang menggunakan peran untuk mengasumsikan peran kedua. Anda tidak dapat mengambil peran saat masuk sebagai Pengguna root akun AWS.

Secara default, sesi peran Anda berlangsung selama satu jam. Saat Anda mengambil peran ini menggunakan operasi AWS STS `AssumeRole*`API, Anda dapat menentukan nilai untuk

`DurationSeconds` parameter tersebut. Nilai ini dapat berkisar dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Jika Anda menggunakan rantai peran, sesi Anda akan dibatasi hingga maksimum satu jam. Jika Anda kemudian menggunakan parameter `DurationSeconds` untuk memberikan nilai lebih dari satu jam, operasi gagal.

> **Note**
>
> Untuk tujuan keamanan, administrator dapat meninjau AWS CloudTrail log untuk mengetahui siapa yang melakukan tindakan. AWS Administrator Anda mungkin akan meminta Anda menentukan identitas sumber atau nama sesi peran ketika Anda mengambil peran tersebut. Untuk informasi selengkapnya, lihat `sts:SourceIdentity` dan `sts:RoleSessionName`.

Contoh kode berikut menunjukkan cara membuat pengguna dan mengambil peran.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat pengguna tanpa izin.
- Buat peran yang memberikan izin untuk mencantumkan bucket Amazon S3 untuk akun tersebut.
- Tambahkan kebijakan agar pengguna dapat mengambil peran tersebut.
- Asumsikan peran dan daftar bucket S3 menggunakan kredensyal sementara, lalu bersihkan sumber daya.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
```

```
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
```

```
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
```

```
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
```

```
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
```

```
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }
```

```
        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
```

```
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
```

```
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
 ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
 ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
```

```
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}
```

```
using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();
```

```
        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
```

```
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");
```

```
        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);
```

```
        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
patience.");
    }
```

```
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
```

```
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
```

```
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
```

```
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }
```

```
    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk .NET .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
```

```
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
```

```
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
```

```
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
```

```
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret

  local buckets
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
  else
    errecho "Because the role with permissions has not been assumed, listing
buckets failed."
  fi

  echo
  echo_repeat "*" 88
  echo "Now assume the role $iam_role_name and list the buckets."
  echo_repeat "*" 88
  echo

  local credentials

  credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
   echo "the assumed role."
 else
   errecho "Failed to list buckets. This should not happen."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   export AWS_SESSION_TOKEN=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 local result=0
 export AWS_ACCESS_KEY_ID=""
 export AWS_SECRET_ACCESS_KEY=""

 echo
 echo_repeat "*" 88
```

```
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Fungsi IAM yang digunakan dalam skenario ini.

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}
```

```
  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name already exists in the account."
  return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
  --output text \
  --query 'User.Arn')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-user operation failed.$response"
  return 1
fi

echo "$response"
```

```
  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_policy"
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
```

```
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
```

```
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

########################################################################################
```

```
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_delete_policy"
  echo "Deletes an AWS Identity and Access Management (IAM) policy"
  echo "  -n policy_arn -- The name of the IAM policy arn."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
```

```
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

########################################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

########################################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
```

```
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
 }

 # Retrieve the calling parameters.
 while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
 done
 export OPTIND=1

 if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
 fi

 iecho "Parameters:\n"
 iecho "    User name:   $user_name"
 iecho ""

 # If the user does not exist, we don't want to try to delete it.
 if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
 fi

 response=$(aws iam delete-user \
    --user-name "$user_name")

 local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
```

```
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;
```

```
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
```

```
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }
```

```
        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
```

```
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());
```

```
        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
```

```
            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }
```

```
        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk C++ .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

Jalankan skenario interaktif di penggugah/prompt perintah.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
```

```
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
```

```
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
```

```
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
```

```
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
```

```
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
```

```
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
```

```
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
*key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
*user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
```

```
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Tentukan struct yang membungkus tindakan akun.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)
```

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Tentukan struct yang membungkus tindakan kebijakan.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
```

```
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
```

```
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan peran.

```
import (
 "context"
 "encoding/json"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
```

```
    Principal: map[string]string{"AWS": trustedUserArn},
    Action:    []string{"sts:AssumeRole"},
   }},
  }
  policyBytes, err := json.Marshal(trustPolicy)
  if err != nil {
   log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
  trustedUserArn, err)
   return nil, err
  }
  result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
   AssumeRolePolicyDocument: aws.String(string(policyBytes)),
   RoleName:                 aws.String(roleName),
  })
  if err != nil {
   log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
  } else {
   role = result.Role
  }
  return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
```

```
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
```

```
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
```

```
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan pengguna.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
```

```
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
```

```
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
```

```
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
```

```
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Go .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan pengguna IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
```

```
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
```

```
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
```

```
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";
```

```
/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
```

```
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
  // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
```

```
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Action: ["s3:ListAllMyBuckets"],
            Resource: "*",
          },
        ],
      }),
```

```
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
```

```
    credentials: {
      accessKeyId: Credentials.AccessKeyId,
      secretAccessKey: Credentials.SecretAccessKey,
      sessionToken: Credentials.SessionToken,
    },
  });

  // List the S3 buckets again.
  // Retry the list buckets operation until it succeeds. AccessDenied might
  // be thrown while the role policy is still stabilizing.
  await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
    listBuckets(s3Client),
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
```

```
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk JavaScript .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan pengguna IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
```

```
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
```

```
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
```

```
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
```

```
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
```

```
    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
```

```
    return jsonParser.parse(reader)
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
```

```
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
```

```
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
```

```
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk PHP .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import json
import sys
```

```
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
```

```
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
```

```
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
```

```
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
```

```
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise



def teardown(user, role):
```

```
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
```

```
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
```

```
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
```

```
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
```

```
 # @return [Aws::IAM::UserPolicy] The newly created policy.
 def create_user_policy(policy_name, user, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 'sts:AssumeRole',
       Resource: role.arn
     }]
   }.to_json
   @iam_client.put_user_policy(
     user_name: user.user_name,
     policy_name: policy_name,
     policy_document: policy_document
   )
   puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
```

```
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
  # @param key_id [String] The ID of the access key used by the STS client.
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end

  # Deletes a role. If the role has policies attached, they are detached and
```

```
  # deleted before the role is deleted.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    @iam_client.list_attached_role_policies(role_name:
 role_name).attached_policies.each do |policy|
      @iam_client.detach_role_policy(role_name: role_name, policy_arn:
 policy.policy_arn)
      @iam_client.delete_policy(policy_arn: policy.policy_arn)
      @logger.info("Detached and deleted policy #{policy.policy_name}.")
    end
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Role deleted: #{role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
```

```
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Ruby .
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;
```

```
    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
```

```
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );
```

```
   let attach_role_policy_result =
       iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
           .await?;
   println!(
       "Attached the policy to the role: {:?}",
       attach_role_policy_result
   );

   let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
   let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
   iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
       .await?;
   println!("Created inline policy.");

   //First, fail to list the buckets with the user.
   let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
   let fail_config = aws_config::from_env()
       .credentials_provider(creds.clone())
       .load()
       .await;
   println!("Fail config: {:?}", fail_config);
   let fail_client: s3Client = s3Client::new(&fail_config);
   match fail_client.list_buckets().send().await {
       Ok(e) => {
           println!("This should not run. {:?}", e);
       }
       Err(e) => {
           println!("Successfully failed with error: {:?}", e)
       }
   }

   let sts_config = aws_config::from_env()
       .credentials_provider(creds.clone())
       .load()
       .await;
   let sts_client: stsClient = stsClient::new(&sts_config);
   sleep(Duration::from_secs(10)).await;
   let assumed_role = sts_client
       .assume_role()
```

```
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
```

```
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy

- DetachRolePolicy
- PutUserPolicy

Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2

Aplikasi yang berjalan pada instans Amazon EC2 harus menyertakan AWS kredenSIAL dalam permintaan AWS API. Anda dapat meminta pengembang menyimpan AWS kredenSIAL langsung di dalam instans Amazon EC2 dan mengizinkan aplikasi dalam contoh tersebut untuk menggunakan kredenSIAL tersebut. Tetapi pengembang kemudian harus mengelola kredenSIAL dan memastikan bahwa mereka dengan aman meneruskan kredenSIAL ke setiap instans dan memperbarui setiap instans Amazon EC2 ketika tiba saatnya untuk memperbarui kredenSIAL. Banyak pekerjaan tambahan.

Sebagai gantinya, Anda dapat dan harus menggunakan peran IAM untuk mengelola kreden SIAL sementara untuk aplikasi yang berjalan pada instans Amazon EC2. Saat menggunakan peran, Anda tidak perlu mendistribusikan kredenSIAL jangka panjang (seperti kredentif masuk atau kunci akses) ke instans Amazon EC2. Sebagai gantinya, peran tersebut memberikan izin sementara yang dapat digunakan aplikasi saat melakukan panggilan ke AWS sumber daya lain. Saat meluncurkan instans Amazon EC2, Anda menentukan peran IAM untuk dikaitkan dengan instans. Aplikasi yang berjalan pada instans kemudian dapat menggunakan kredensial sementara yang diberikan oleh peran untuk menandatangani permintaan API.

Menggunakan peran untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2 memerlukan sedikit konfigurasi tambahan. Aplikasi yang berjalan pada instans Amazon EC2 disabstrahkan AWS oleh sistem operasi virtual. Karena pemisahan ekstra ini, Anda memerlukan langkah tambahan untuk menetapkan AWS peran dan izin terkaitnya ke instans Amazon EC2 dan membuatnya tersedia untuk aplikasinya. Langkah tambahan ini adalah pembuatan profil instance yang dil ampirkan ke instance. Profil instans berisi peran dan dapat memberikan kredensial sementara peran ke aplikasi yang berjalan pada instans. Kredensial sementara tersebut kemudian dapat digunakan dalam panggilan API aplikasi untuk mengakses sumber daya dan membatasi akses ke sumber daya yang ditentukan oleh peran tersebut saja.

> **Note**
>
> Hanya satu peran yang dapat ditetapkan ke instans Amazon EC2 pada satu waktu, dan semua aplikasi pada instans berbagi peran dan izin yang sama. Saat memanfaatkan

Amazon ECS untuk mengelola instans Amazon EC2, Anda dapat menetapkan peran ke tugas Amazon ECS yang dapat dibedakan dari peran instans Amazon EC2 yang dijalankan. Menetapkan setiap tugas peran sejalan dengan prinsip akses yang paling tidak istimewa dan memungkinkan kontrol terperinci yang lebih besar atas tindakan dan sumber daya. Untuk informasi selengkapnya, lihat Menggunakan peran IAM dengan tugas Amazon ECS di Panduan Praktik Terbaik Layanan Kontainer Amazon Elastic.

Penggunaan peran dengan cara ini memiliki beberapa keuntungan. Karena kredenSIAL peran bersifat sementara dan diperbarui secara otomatis, Anda tidak perlu mengelola kredenSIAL, dan Anda tidak perlu khawatir tentang risiko keamanan jangka panjang. Selain itu, jika Anda menggunakan peran tunggal untuk beberapa instance, Anda dapat membuat perubahan pada satu peran tersebut dan perubahan tersebut merambat secara otomatis ke semua instance.

Note

Meskipun peran biasanya ditugaskan ke instans Amazon EC2 saat Anda meluncurkannya, peran juga dapat dilampirkan ke instans Amazon EC2 yang sedang berjalan. Untuk mempelajar cara melampirkan peran pada instans yang sedang berjalan, lihat Peran IAM untuk Amazon EC2.

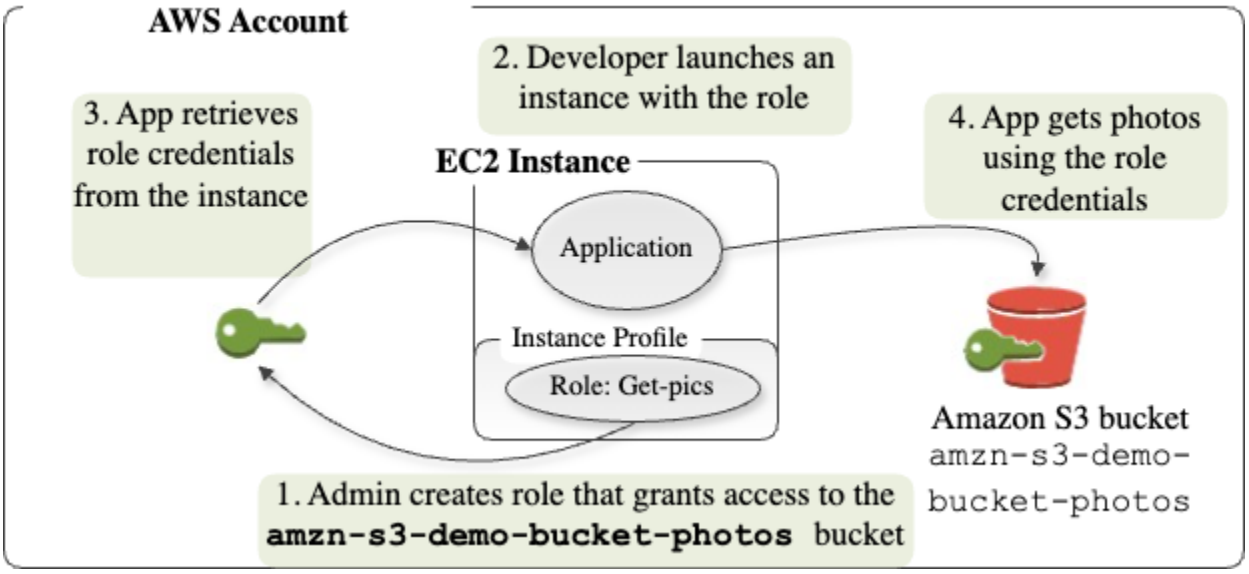
Topik

- Bagaimana cara kerja peran untuk instans Amazon EC2?
- Izin yang diperlukan untuk menggunakan peran dengan Amazon EC2
- Bagaimana saya memulainya?
- Informasi Terkait

Bagaimana cara kerja peran untuk instans Amazon EC2?

Pada gambar berikut, pengembang menjalankan aplikasi pada instans Amazon EC2 yang memerlukan akses ke bucket S3 bernama`aamzn-s3-demo-bucket-photos`. Administrator membuat peran `Get-pics` layanan dan melampirkan peran tersebut ke instans Amazon EC2. Peran ini mencakup kebijakan izin yang memberikan akses hanya-baca ke bucket S3 tertentu. Ini juga mencakup kebijakan kepercayaan yang memungkinkan instans Amazon EC2 untuk mengambil peran dan mengambil kredenSIAL sementara. Saat aplikasi berjalan pada instans, itu menggunakan

kredensial sementara peran untuk mengakses bucket foto. Administrator tidak perlu memberikan izin kepada developer untuk mengakses bucket foto, dan developer tidak perlu membagikan atau mengelola kredensial.



1. Administrator menggunakan IAM untuk membuat peran **`Get-pics`**. Dalam kebijakan kepercayaan peran, administrator menentukan bahwa hanya instans Amazon EC2 yang dapat mengambil peran tersebut. Dalam kebijakan izin peran, administrator menentukan izin hanya-baca untuk bucket `amzn-s3-demo-bucket-photos`.
2. Pengembang meluncurkan instans Amazon EC2 dan menetapkan `Get-pics` peran ke instance tersebut.

   > **Note**
   >
   > Jika Anda menggunakan konsol IAM, profil instans dikelola untuk Anda dan sebagian besar transparan bagi Anda. Namun, jika Anda menggunakan API AWS CLI or untuk membuat dan mengelola peran dan instans Amazon EC2, maka Anda harus membuat profil instans dan menetapkan peran tersebut sebagai langkah terpisah. Kemudian, saat Anda meluncurkan instans, Anda harus menentukan nama profil instans dan bukan nama peran.

3. Saat aplikasi berjalan, itu mendapatkan kredensial keamanan sementara dari metadata instans Amazon EC2, sebagaimana dijelaskan dalam Mengambil Kredensial Keamanan dari Metadata Instans. Ini adalah kredensial keamanan sementara yang mewakili peran tersebut dan berlaku untuk periode waktu yang terbatas.

Dengan beberapa SDK AWS, developer dapat menggunakan penyedia yang mengelola kredensial keamanan sementara secara transparan. (Dokumentasi untuk AWS SDK individual menjelaskan fitur yang didukung oleh SDK tersebut untuk mengelola kredenSIAL.)

Atau, aplikasi bisa mendapatkan kredenSIAL sementara langsung dari metadata instans Amazon EC2. Kredensial dan nilai terkait tersedia dari kategori `iam/security-credentials/`*`role-name`* (dalam hal ini, `iam/security-credentials/Get-pics`) metadata. Jika aplikasi mendapatkan kredensial dari metadata instans, itu dapat menyimpan kredensial tersebut.

4. Dengan menggunakan kredensial sementara yang diambil, aplikasi mengakses bucket foto. Karena kebijakan terlampir pada peran **`Get-pics`**, aplikasi memiliki izin hanya-baca.

   KredenSIAL keamanan sementara yang tersedia di instans diperbarui secara otomatis sebelum kedaluwarsa sehingga kumpulan yang valid selalu tersedia. Aplikasi hanya perlu memastikan bahwa itu mendapatkan serangkaian kredensial baru dari metadata instans sebelum metadata yang sedang digunakan kedaluwarsa. Dimungkinkan untuk menggunakan AWS SDK untuk mengelola kredenSIAL sehingga aplikasi tidak perlu menyertakan logika tambahan untuk menyegarkan kredenSIAL. Misalnya, membuat instan klien dengan Penyedia Kredensial Profil Instans. Namun, jika aplikasi mendapatkan kredensial keamanan sementara dari metadata instans dan sudah menyimpannya, itu akan mendapatkan rangkaian kredensial yang diperbarui setiap jam, atau setidaknya 15 menit sebelum rangkaian yang sedang digunakan kedaluwarsa. Waktu kedaluwarsa termasuk dalam informasi yang dikembalikan dalam `iam/security-credentials/`*`role-name`* kategori.

Izin yang diperlukan untuk menggunakan peran dengan Amazon EC2

Untuk meluncurkan instans dengan peran, pengembang harus memiliki izin untuk meluncurkan instans Amazon EC2 dan izin untuk meneruskan peran IAM.

Kebijakan sampel berikut memungkinkan pengguna untuk menggunakan Konsol Manajemen AWS untuk meluncurkan instance dengan peran. Kebijakan ini menyertakan wildcard (*) untuk mengizinkan pengguna meneruskan peran apa pun dan melakukan tindakan Amazon EC2 yang terdaftar. T `ListInstanceProfiles` indakan ini memungkinkan pengguna untuk melihat semua peran yang tersedia di Akun AWS.

Example Contoh kebijakan yang memberikan izin kepada pengguna untuk menggunakan konsol Amazon EC2 untuk meluncurkan instans dengan peran.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Membatasi peran mana yang dapat diteruskan ke instans Amazon EC2 (menggunakan PassRole)

Anda dapat menggunakan `PassRole` izin untuk membatasi peran mana yang dapat diteruskan pengguna ke instans Amazon EC2 saat pengguna meluncurkan instans. Ini membantu mencegah pengguna menjalankan aplikasi yang memiliki izin lebih dari yang diberikan kepada pengguna—yaitu, agar dapat memperoleh hak istimewa yang ditingkatkan. Misalnya, bayangkan pengguna Alice hanya memiliki izin untuk meluncurkan instans Amazon EC2 dan bekerja dengan bucket Amazon

S3, tetapi peran yang ia berikan ke instans Amazon EC2 memiliki izin untuk bekerja dengan IAM dan Amazon DynamoDB. Dalam hal ini, Alice mungkin dapat meluncurkan instans, masuk ke dalamnya, mendapatkan kredenSIAL keamanan sementara, dan kemudian melakukan tindakan IAM atau DynamoDB yang tidak diotorisasi.

Untuk membatasi peran mana yang dapat diteruskan pengguna ke instans Amazon EC2, Anda membuat kebijakan yang mengizinkan `PassRole` tindakan tersebut. Anda kemudian melampirkan kebijakan ke pengguna (atau ke grup IAM milik pengguna) yang akan meluncurkan instans Amazon EC2. Di `Resource` elemen kebijakan, Anda mencantumkan peran atau peran yang diizinkan untuk diteruskan pengguna ke instans Amazon EC2. Saat pengguna meluncurkan suatu instans dan mengaitkan peran dengannya, Amazon EC2 akan memeriksa apakah pengguna diizinkan untuk meneruskan peran tersebut. Tentu saja, Anda juga harus memastikan bahwa peran yang dapat diteruskan oleh pengguna tidak mencakup lebih banyak izin dari yang seharusnya dimiliki oleh pengguna.

> **Note**
>
> `PassRole` bukan merupakan tindakan API dengan cara yang sama seperti `RunInstances` atau `ListInstanceProfiles`. Sebaliknya, ini adalah izin yang AWS memeriksa setiap kali peran ARN diteruskan sebagai parameter ke API (atau konsol melakukan ini atas nama pengguna). Itu membantu administrator untuk mengontrol peran mana yang dapat diteruskan oleh pengguna yang mana. Dalam kasus ini, itu memastikan bahwa pengguna diizinkan untuk melampirkan peran tertentu ke instans Amazon EC2.

Example Contoh kebijakan yang memberikan izin pengguna untuk meluncurkan instans Amazon EC2 dengan peran tertentu

Kebijakan sampel berikut memungkinkan pengguna untuk menggunakan Amazon EC2 API untuk meluncurkan instans dengan peran. Elemen `Resource` menentukan Amazon Resource Name (ARN) dari peran. Dengan menentukan ARN, kebijakan memberi izin kepada pengguna untuk hanya meneruskan peran `Get-pics`. Jika pengguna mencoba menentukan peran yang berbeda saat meluncurkan suatu instans, tindakan tersebut gagal. Pengguna memang memiliki izin untuk menjalankan setiap instans, terlepas dari apakah mereka meneruskan suatu peran.

JSON

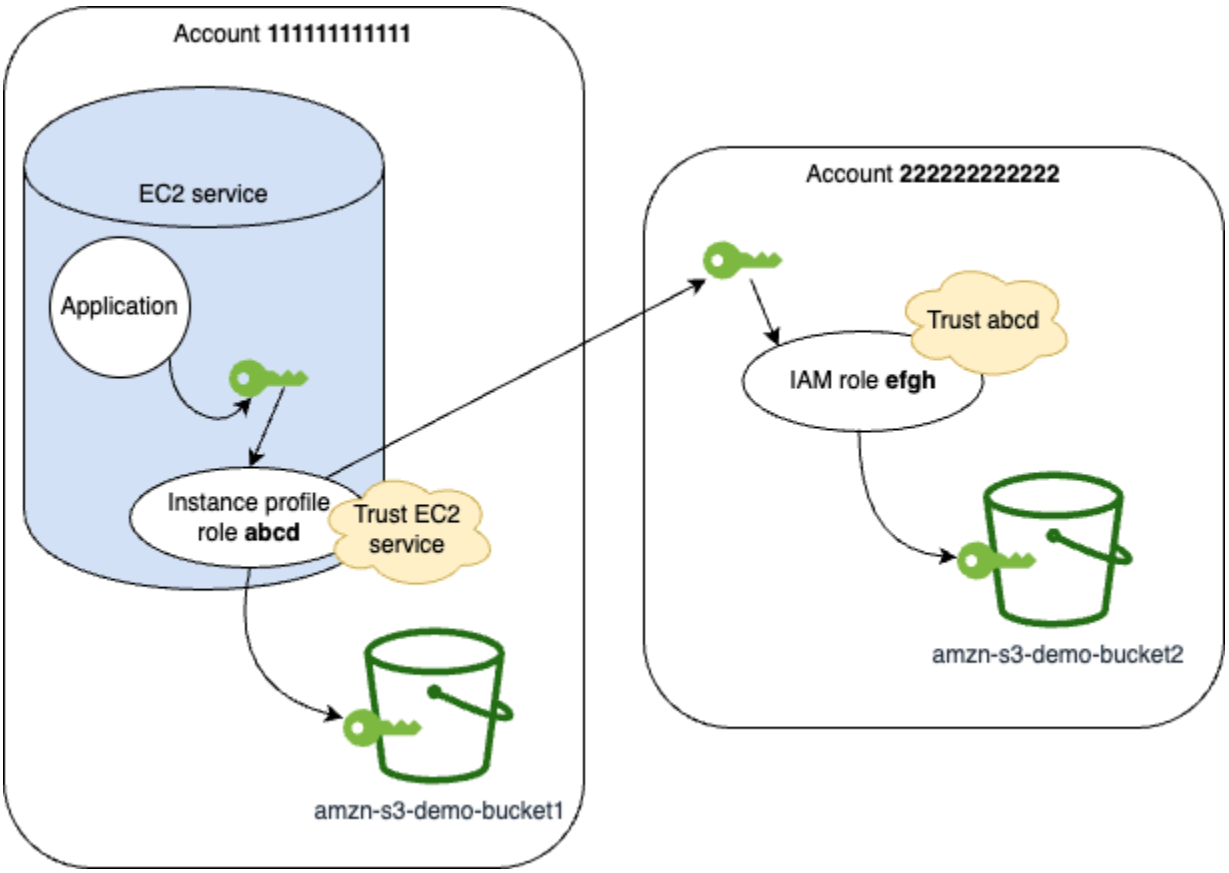
```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/Get-pics"
        }
    ]
}
```

Memungkinkan peran profil instans untuk beralih ke peran dalam akun lain

Anda dapat mengizinkan aplikasi yang berjalan di instans Amazon EC2 untuk menjalankan perintah di akun lain. Untuk melakukan ini, Anda harus mengizinkan peran instans Amazon EC2 di akun pertama untuk beralih ke peran di akun kedua.

Bayangkan Anda menggunakan dua Akun AWS dan Anda ingin mengizinkan aplikasi yang berjalan pada instans Amazon EC2 untuk menjalankan AWS CLI perintah di kedua akun. Asumsikan bahwa instance Amazon EC2 ada di akun`111111111111`. Instance tersebut menyertakan peran profil `abcd` instance yang memungkinkan aplikasi melakukan tugas Amazon S3 yang hanya dapat dibaca pada `amzn-s3-demo-bucket1` bucket dalam akun yang sama`111111111111`. Namun demikian, aplikasi tersebut juga harus diizinkan untuk menjalankan `efgh` peran lintas akun untuk mengakses `amzn-s3-demo-bucket2` bucket Amazon S3 dalam akun `222222222222`.



Per `abcd` an profil instans Amazon EC2 harus memiliki kebijakan izin berikut untuk mengizinkan aplikasi mengakses bucket `amzn-s3-demo-bucket1` Amazon S3:

Akun 111111111111 Kebijakan Izin Peran ***abcd***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
```

```
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Peran abcd harus mempercayai layanan Amazon EC2 untuk menjalankan peran tersebut. Untuk melakukannya, peran abcd harus memiliki kebijakan kepercayaan berikut:

Akun 111111111111 Kebijakan Kepercayaan Peran ***abcd***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}
```

Anggap bahwa peran lintas akun `efgh` memungkinkan tugas Amazon S3 hanya baca di bucket `amzn-s3-demo-bucket2` dalam akun `222222222222` yang sama. Untuk melakukannya, peran lintas akun `efgh` harus memiliki kebijakan izin berikut:

Akun 222222222222 Kebijakan Izin Peran ***efgh***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

Peran `efgh` harus mempercayai profil instans `abcd` untuk menjalankannya. Untuk melakukannya, peran `efgh` harus memiliki kebijakan kepercayaan berikut:

Akun 222222222222 Kebijakan Kepercayaan Peran ***efgh***

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Bagaimana saya memulainya?

Untuk memahami cara kerja peran dengan instans Amazon EC2, Anda perlu menggunakan konsol IAM untuk membuat peran, meluncurkan instans Amazon EC2 yang menggunakan peran tersebut, lalu memeriksa instance yang sedang berjalan. Anda dapat menguji metadata instans untuk melihat bagaimana kredensial sementara peran tersedia untuk sebuah instans. Anda juga dapat melihat bagaimana aplikasi yang berjalan pada suatu instans dapat menggunakan peran tersebut. Gunakan sumber daya berikut ini untuk mempelajari lebih banyak.

-
- Penelusuran SDK. Dokumentasi AWS SDK menyertakan panduan yang menunjukkan aplikasi yang berjalan pada instans Amazon EC2 yang menggunakan kredentif sementara untuk peran untuk membaca bucket Amazon S3. Setiap penelusuran berikut menyajikan langkah serupa dengan bahasa pemrograman yang berbeda:
  - Konfigurasikan Peran IAM untuk Amazon EC2 dengan SDK untuk Java di Panduan Pengembang AWS SDK untuk Java
  - Meluncurkan Instans Amazon EC2 menggunakan SDK untuk.NET di Panduan Peng AWS SDK untuk .NET embang
  - Membuat Instans Amazon EC2 dengan SDK untuk Ruby di Panduan Peng AWS SDK untuk Ruby embang

Informasi Terkait

Untuk informasi selengkapnya tentang membuat peran atau peran untuk instans Amazon EC2, lihat informasi berikut:

- Untuk informasi selengkapnya tentang menggunakan peran IAM dengan instans Amazon EC2, buka Panduan Pengguna Amazon EC2.
- Untuk membuat peran, lihat Pembuatan peran IAM
- Untuk informasi lebih lanjut tentang membuat kredensial keamanan sementara, lihat Kredensial keamanan sementara di IAM.
- Jika Anda bekerja dengan IAM API atau CLI, Anda harus membuat dan mengelola profil instans IAM. Untuk informasi selengkapnya tentang profil instans, lihat Gunakan profil instance.
- Untuk informasi selengkapnya tentang kredenSIAL keamanan sementara untuk peran dalam metadata instans, lihat Mengambil KredenSIAL Keamanan dari Metadata Instans di Panduan Pengguna Amazon EC2.

Gunakan profil instance

Gunakan profil instans untuk meneruskan peran IAM ke instans EC2. Untuk informasi selengkapnya, lihat peran IAM untuk Amazon EC2 di Panduan Pengguna Amazon EC2.

Mengelola profil instans (konsol)

Jika Anda menggunakan Konsol Manajemen AWS untuk membuat peran untuk Amazon EC2, konsol secara otomatis membuat profil instance dan memberinya nama yang sama dengan peran tersebut. Ketika Anda kemudian menggunakan konsol Amazon EC2 untuk meluncurkan suatu instans dengan peran IAM, Anda dapat memilih peran untuk dikaitkan dengan isntans. Dalam konsol, daftar yang ditampilkan sebenarnya adalah daftar nama profil instance. Konsol tidak membuat profil instans untuk peran yang tidak terkait dengan Amazon EC2.

Anda dapat menggunakan Konsol Manajemen AWS untuk menghapus peran IAM dan profil instans untuk Amazon EC2 jika peran dan profil instans memiliki nama yang sama. Untuk mempelajari selengkapnya tentang menghapus profil instans, lihat Hapus peran atau profil contoh.

> **Note**
>
> Untuk memperbarui izin untuk sebuah instance, ganti profil instansnya. Kami tidak menyarankan untuk menghapus peran dari profil instance, karena ada penundaan hingga satu jam sebelum perubahan ini berlaku.

Mengelola profil instance (AWS CLI atau AWS API)

Jika Anda mengelola peran dari AWS CLI atau AWS API, Anda membuat profil peran dan instance sebagai tindakan terpisah. Karena peran dan profil instans dapat memiliki nama yang berbeda, Anda harus mengetahui nama profil instans Anda serta nama peran yang dimuat olehnya. Dengan begitu Anda dapat memilih profil instans yang benar saat Anda meluncurkan instans EC2.

Anda dapat melampirkan tag ke sumber daya IAM Anda, termasuk profil instance, untuk mengidentifikasi, mengatur, dan mengontrol akses ke sumber daya tersebut. Anda dapat menandai profil instance hanya saat Anda menggunakan AWS API AWS CLI or.

> **Note**
>
> Profil instans hanya dapat berisi satu peran IAM, meskipun peran dapat dimasukkan dalam beberapa profil instans. Batasan satu peran per profil instans ini tidak dapat ditingkatkan. Anda dapat menghapus peran yang ada dan kemudian menambahkan peran yang berbeda ke profil instans. Anda kemudian harus menunggu perubahan muncul di semua AWS karena konsist ensi akhirnya. Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Mengelola profil instance (AWS CLI)

Anda dapat menggunakan AWS CLI perintah berikut untuk bekerja dengan profil instance di AWS akun.

- Buat profil instans: `aws iam create-instance-profile`
- Tandai profil instans: `aws iam tag-instance-profile`
- Cantumkan tanda untuk profil instans: `aws iam list-instance-profile-tags`
- Hapus tanda profil instans: `aws iam untag-instance-profile`

- Tambahkan peran ke profil instans: `aws iam add-role-to-instance-profile`
- Buat daftar profil instans: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Dapatkan informasi tentang profil instans: `aws iam get-instance-profile`
- Hapus peran dari profil instans: `aws iam remove-role-from-instance-profile`
- Hapus profil instans: `aws iam delete-instance-profile`

Anda juga dapat melampirkan peran ke instans EC2 yang sudah berjalan dengan menggunakan perintah berikut. Untuk informasi lebih lanjut, lihat Peran IAM untuk Amazon EC2.

- Lampirkan profil instans dengan peran ke instans EC2 yang dihentikan atau sedang berjalan: `aws ec2 associate-iam-instance-profile`
- Dapatkan informasi tentang profil instans yang terlampir pada instans EC2: `aws ec2 describe-iam-instance-profile-associations`
- Lepas profil instans dengan peran dari instans EC2 yang dihentikan atau sedang berjalan: `aws ec2 disassociate-iam-instance-profile`

Mengelola profil instance (AWS API)

Anda dapat memanggil operasi AWS API berikut untuk bekerja dengan profil instance di Akun AWS.

- Buat profil instans: `CreateInstanceProfile`
- Tandai profil instans: `TagInstanceProfile`
- Cantumkan tanda di profil instans: `ListInstanceProfileTags`
- Hapus tanda profil instans: `UntagInstanceProfile`
- Tambahkan peran ke profil instans: `AddRoleToInstanceProfile`
- Buat daftar profil instans: `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Dapatkan informasi tentang profil instans: `GetInstanceProfile`
- Hapus peran dari profil instans: `RemoveRoleFromInstanceProfile`
- Hapus profil instans: `DeleteInstanceProfile`

Anda juga dapat melampirkan peran ke instans EC2 yang sudah berjalan dengan memanggil operasi berikut ini. Untuk informasi lebih lanjut, lihat Peran IAM untuk Amazon EC2.

- Lampirkan profil instans dengan peran ke instans EC2 yang dihentikan atau sedang berjalan: `AssociateIamInstanceProfile`
- Dapatkan informasi tentang profil instans yang terlampir pada instans EC2: `DescribeIamInstanceProfileAssociations`
- Lepas profil instans dengan peran dari instans EC2 yang dihentikan atau sedang berjalan: `DisassociateIamInstanceProfile`

Penyedia identitas dan federasi ke AWS

Sebagai praktik terbaik, sebaiknya Anda mewajibkan pengguna manusia menggunakan federasi dengan penyedia identitas untuk mengakses AWS sumber daya alih-alih membuat pengguna IAM individual di Anda Akun AWS. Dengan penyedia identitas (IdP), Anda dapat mengelola identitas pengguna di luar AWS dan memberikan izin identitas pengguna eksternal ini untuk menggunakan AWS sumber daya di akun Anda. Ini berguna jika organisasi Anda sudah memiliki sistem identitas sendiri, seperti direktori pengguna korporat. Ini juga berguna jika Anda membuat aplikasi seluler atau aplikasi web yang membutuhkan akses ke AWS sumber daya.

> **Note**
>
> Anda juga dapat mengelola pengguna manusia di Pusat Identi tas IAM dengan penyedia identitas SAML eksternal alih-alih menggunakan federasi SAML di IAM. Federasi Pusat Identitas IAM dengan penyedia identitas menyediakan kemampuan bagi Anda untuk memberi orang akses ke beberapa AWS akun di organisasi Anda dan ke beberapa AWS aplikasi. Untuk informasi tentang situasi tertentu di mana pengguna IAM diperlukan, lihat Kapan membuat pengguna IAM (bukan peran).
>
> Anda juga dapat menggunakan manajer akses akun — fitur IAM yang memungkinkan Anda menetapkan peran IAM yang ada di seluruh akun organisasi Anda kepada pengguna dan grup Pusat Identitas IAM. Anda dapat menggunakannya bersama kumpulan izin atau sendiri.

Jika Anda lebih suka menggunakan satu AWS akun tanpa mengaktifkan Pusat Identitas IAM, Anda dapat menggunakan IAM dengan IdP eksternal yang menyediakan informasi identitas untuk AWS menggunakan OpenID Connect (OIDC) atau SAML 2.0 (Security Assertion Markup Language 2.0). OIDC menghubungkan aplikasi, seperti T GitHub indakan, yang tidak berjalan AWS ke AWS sumber daya. Contoh penyedia identitas SAML yang terkenal adalah Shibboleth dan Active Directory Federation Services.

Saat Anda menggunakan penyedia identitas , Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. IdP menyediakannya untuk Anda. Pengguna eksternal Anda masuk melalui IdP, dan Anda dapat memberikan izin identitas eksternal tersebut untuk menggunakan AWS sumber daya di akun Anda. Penyedia identitas membantu menjaga keamanan Akun AWS Anda karena Anda tidak perlu mendistribusikan atau menyematkan kredentif keamanan jangka panjang, seperti kunci akses, di aplikasi Anda.

Tinjau tabel berikut untuk membantu menentukan jenis federasi IAM mana yang terbaik untuk kasus penggunaan Anda; IAM, Pusat Identitas IAM, atau Amazon Cognito. Ringkasan dan tabel berikut memberikan gambaran umum tentang metode yang dapat digunakan pengguna Anda untuk mendapatkan akses federasi ke sumber daya. AWS

Jenis federasi IAM	Jenis akun	Manajemen akses..	Sumber identitas yang didukung
Federasi dengan Pusat Identitas IAM	Beberapa akun dikelola oleh AWS Organizations	Pengguna manusia tenaga kerja Anda	• SAML 2.0 • Direktori Aktif Terkelola • Direktori Identity Center
Federasi dengan IAM	Akun tunggal, mandiri	• Pengguna manusia dalam penyebaran jangka pendek dan skala kecil • Pengguna mesin	• SAML 2.0 • OIDC
Federasi dengan kumpulan identitas Amazon Cognito	Setiap	Pengguna aplikasi yang memerlukan otorisasi IAM untuk mengakses sumber daya	• SAML 2.0 • OIDC • Pilih penyedia identitas sosial OAuth 2.0

Federasi dengan Pusat Identitas IAM

Untuk pengelolaan akses terpusat pengguna manusia, sebaiknya gunakan Pusat Identi tas IAM untuk mengelola akses ke akun dan izin dalam akun tersebut. Pengguna di Pusat Identitas IAM diberikan kredensi jangka pendek ke AWS sumber daya Anda. Anda dapat menggunakan Active Directory, penyedia identitas eksternal (IdP), atau direktori Pusat Identitas IAM sebagai sumber identitas bagi pengguna dan grup untuk menetapkan akses ke sumber daya Anda AWS .

IAM Identity Center mendukung federasi identitas dengan SAML (Security Assertion Markup Language) 2.0 untuk menyediakan akses masuk tunggal federasi bagi pengguna yang berwenang untuk menggunakan aplikasi dalam portal akses. AWS Pengguna kemudian dapat masuk tunggal ke layanan yang mendukung SAML, termasuk aplikasi pihak ketiga, seperti Microsoft 365, SAP Concur, Konsol Manajemen AWS dan Salesforce.

Federasi dengan IAM

Meskipun kami sangat menyarankan mengelola pengguna manusia di Pusat Identitas IAM, Anda dapat mengaktifkan akses utama federasi dengan IAM untuk pengguna manusia dalam penerapan skala kecil jangka pendek. IAM memungkinkan Anda untuk menggunakan SAML 2.0 dan Open ID Connect (OIDC) terpisah IdPs dan menggunakan atribut utama federasi untuk kontrol akses. Dengan IAM, Anda dapat meneruskan atribut pengguna, seperti pusat biaya, judul, atau lokal, dari IdPs ke AWS, dan menerapkan izin akses berbutir halus berdasarkan atribut ini.

B eban kerja adalah kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi atau proses backend. Beban kerja Anda mungkin memerlukan identitas IAM untuk membuat permintaan ke AWS layanan, aplikasi, alat operasional, dan komponen. Identitas ini mencakup mesin yang berjalan di AWS lingkungan Anda, seperti instans atau AWS Lambda fungsi Amazon EC2.

Anda juga dapat mengelola identitas mesin untuk pihak eksternal yang membutuhkan akses. Untuk memberikan akses ke identitas mesin, Anda dapat menggunakan peran IAM. Peran IAM memiliki izin khusus dan menyediakan cara untuk mengakses AWS dengan mengandalkan kredentif keamanan sementara dengan sesi peran. Selain itu, Anda mungkin memiliki mesin di luar AWS yang membutuhkan akses ke AWS lingkungan Anda. Untuk mesin yang berjalan di luar AWS Anda dapat menggunakan IAM Roles Anywhere. Untuk informasi lebih lanjut tentang peran, lihat Peran IAM. Untuk detail tentang cara menggunakan peran untuk mendelegasikan akses Akun AWS, lihatTutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM.

Untuk menautkan IdP secara langsung ke IAM, Anda membuat entitas penyedia identitas untuk membangun hubungan kepercayaan antara Anda Akun AWS dan IdP. Dukungan IAM IdPs yang

kompatibel dengan OpenID Connect (OIDC) atau SAML 2.0 (Security Assertion Markup Language 2.0). Untuk informasi selengkapnya tentang menggunakan salah satu dari ini IdPs dengan AWS, lihat bagian berikut:

- Federasi OIDC
- Federasi SAML 2.0

Federasi dengan kumpulan identitas Amazon Cognito

Amazon Cognito dirancang untuk pengembang yang ingin mengotentikasi dan mengotorisasi pengguna di aplikasi seluler dan web mereka. Kumpulan pengguna Amazon Cognito menambahkan fitur masuk dan pendaftaran ke aplikasi Anda, dan kumpulan identitas memberikan kredentif IAM yang memberi pengguna akses ke sumber daya terlindungi yang Anda kelola. AWS Kumpulan identitas memperoleh kredenSIAL untuk sesi sementara melalui operasi `AssumeRoleWithWebIdentity` API.

Amazon Cognito bekerja dengan penyedia identitas eksternal yang mendukung SAML dan OpenID Connect, dan dengan penyedia identitas sosial seperti Facebook, Google, dan Amazon. Aplikasi Anda dapat masuk ke pengguna dengan kumpulan pengguna atau IdP eksternal, lalu mengambil sumber daya atas nama mereka dengan sesi sementara yang disesuaikan dalam peran IAM.

Sumber daya tambahan

- Untuk demonstrasi tentang cara membuat proxy federasi kustom yang mengaktifkan single sign-on (SSO) ke dalam Konsol Manajemen AWS menggunakan sistem otentikasi organisasi Anda, lihat. Aktifkan akses broker identitas khusus ke AWS konsol
- Untuk petunjuk tentang cara mengaktifkan AWS beban kerja Anda untuk mengakses layanan eksternal tanpa menyimpan kredenSIAL, lihatFederasi AWS Identitas ke layanan eksternal.

Skenario umum

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia untuk menggunakan kredenSIAL sementara saat mengakses AWS. Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk secara terpusat mengelola akses ke beberapa Akun AWS dan menyediakan akses masuk MFA-

protected tunggal kepada pengguna ke semua akun yang ditugaskan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di Pusat Identitas IAM atau dengan mudah terhubung ke penyedia identitas yang kompatibel dengan SAML 2.0 yang ada. Untuk informasi selengkapnya, lihat Apa itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Anda dapat menggunakan penyedia identitas eksternal (IdP) untuk mengelola identitas pengguna di luar AWS dan IdP eksternal. IdP eksternal dapat memberikan informasi identitas untuk AWS menggunakan OpenID Connect (OIDC) atau Security Assertion Markup Language (SAML). OIDC umumnya digunakan ketika aplikasi yang tidak berjalan AWS membutuhkan akses ke AWS sumber daya.

Bila Anda ingin mengonfigurasi federasi dengan IdP eksternal, Anda membuat penyedia identitas IAM untuk menginform asikan AWS tentang IdP eksternal dan konfigurasinya. Ini membangun kepercayaan antara Anda Akun AWS dan IdP eksternal. Topik berikut menyediakan skenario umum untuk menggunakan penyedia identitas IAM.

Topik

- Amazon Cognito untuk aplikasi seluler
- Federasi OIDC untuk aplikasi seluler

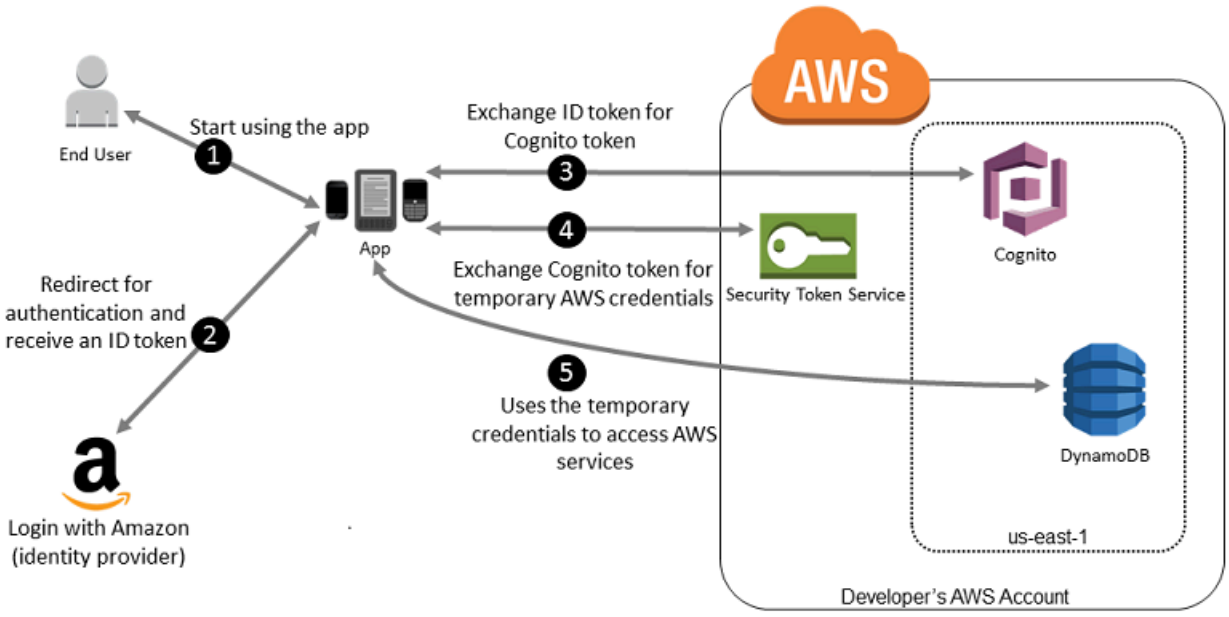
Amazon Cognito untuk aplikasi seluler

Cara yang lebih disukai untuk menggunakan federasi OIDC adalah dengan menggunakan Amazon Cognito. Misalnya, Adele sang pengembang sedang membangun game untuk perangkat seluler di mana data pengguna seperti skor dan profil disimpan di Amazon S3 dan Amazon DynamoDB. Adele juga dapat menyimpan data ini secara lokal di perangkat dan menggunakan Amazon Cognito agar tetap disinkronkan di seluruh perangkat. Ia tahu bahwa untuk alasan keamanan dan pemeliharaan, kredensial keamanan jangka panjang AWS tidak boleh didistribusikan bersama dengan game. Dia juga tahu bahwa game akan memiliki banyak sekali pengguna. Untuk semua alasan ini, dia tidak ingin membuat identitas pengguna baru di IAM untuk setiap pemain. Alih-alih, dia membangun game agar pengguna dapat masuk menggunakan identitas yang sudah mereka tetapkan dengan penyedia identitas eksternal (IdP) yang terkenal, seperti Login with Amazon, Facebook, Google, atau IdP yang kompatibel dengan OpenID Connect (OIDC). Game ini bisa memanfaatkan mekanisme autentikasi dari salah satu penyedia ini untuk memvalidasi identitas pengguna.

Untuk mengaktifkan aplikasi seluler mengakses AWS sumber dayanya, Adele pertama-tama mendaftar untuk ID pengembang dengan pilihannya IdPs. Dia juga mengonfigurasi aplikasi dengan setiap penyedia ini. Di dalamnya Akun AWS yang berisi bucket Amazon S3 dan tabel DynamoDB untuk game tersebut, Adele menggunakan Amazon Cognito untuk membuat peran IAM yang secara tepat menentukan izin yang dibutuhkan game. Jika dia menggunakan IDC IdP, dia juga membuat entitas penyedia identitas IAM OIDC untuk membangun kepercayaan antara kumpulan identitas Amazon Cognito di dirinya dan IdP. Akun AWS

Dalam kode aplikasi, Adele memanggil antarmuka masuk untuk IdP yang sudah dikonfigurasi sebelumnya. IdP menangani semua detail yang memungkinkan pengguna masuk, dan aplikasi mendapatkan akses token OAuth atau token ID OIDC dari penyedia. Aplikasi Adele dapat menukar informasi otentikasi ini dengan satu set kredenSIAL keamanan sementara yang terdiri dari ID kunci AWS akses, kunci akses rahasia, dan token sesi. Aplikasi kemudian dapat menggunakan kredenSIAL ini untuk mengakses layanan web yang ditawarkan oleh AWS. Aplikasi ini terbatas dengan izin yang ditentukan dalam peran yang sudah diasumsikan.

Gambar berikut menunjukkan alur cara kerja yang disederhanakan, menggunakan Login with Amazon sebagai IdP. Untuk Langkah 2, aplikasi juga dapat menggunakan Facebook, Google, atau OIDC-compatible IdP apa pun, tetapi itu tidak ditampilkan di sini.



1. Seorang pelanggan memulai aplikasi Anda di perangkat seluler. Aplikasi meminta pengguna untuk masuk.
2. Aplikasi menggunakan sumber daya Login with Amazon untuk menerima kredensial pengguna.

3. Aplikasi ini menggunakan operasi Amazon Cognito API `GetId` dan `GetCredentialsForIdentity` untuk menukar Login dengan token ID Amazon dengan token Amazon Cognito. Amazon Cognito, yang telah dikonfigurasi untuk mempercayai proyek Login dengan Amazon Anda, menghasilkan token yang ditukar dengan AWS STS kredenSIAL sesi sementara.
4. Aplikasi menerima kredenSIAL keamanan sementara dari Amazon Cognito. Aplikasi Anda juga dapat menggunakan alur kerja Dasar (Klasik) di Amazon Cognito untuk mengambil token dari AWS STS penggunaan`AssumeRoleWithWebIdentity`. Untuk informasi selengkapnya, lihat Alu r otentikasi kumpulan identitas (identitas federasi) di Panduan Pengembang Amazon Cognito.
5. Kredensial keamanan sementara dapat digunakan oleh aplikasi untuk mengakses setiap sumber daya AWS yang diperlukan oleh aplikasi untuk beroperasi. Peran yang terkait dengan kredentif keamanan sementara dan kebijakan yang ditetapkan menentukan apa yang dapat diakses.

Gunakan proses berikut untuk mengonfigurasi aplikasi Anda agar menggunakan Amazon Cognito untuk mengotentikasi pengguna dan memberikan aplikasi Anda akses ke AWS sumber daya. Untuk langkah-langkah khusus untuk menyelesaikan skenario ini, lihat dokumentasi untuk Amazon Cognito.

1. (Opsional) Mendaftar sebagai pengembang dengan Login dengan Amazon, Facebook, Google, atau IdP yang kompatibel dengan OpenID Connect (OIDC) lainnya dan konfigurasikan satu atau lebih aplikasi dengan penyedia. Langkah ini bersifat opsional karena Amazon Cognito juga mendukung akses tidak terotentikasi (tamu) untuk pengguna Anda.
2. Buka Amazon Cognito di Konsol Manajemen AWS. Gunakan wizard Amazon Cognito untuk membuat kumpulan identitas, yang merupakan wadah yang digunakan Amazon Cognito untuk menjaga identitas pengguna akhir tetap tertata untuk aplikasi Anda. Anda dapat berbagi kolam identitas di antara aplikasi. Saat Anda menyiapkan kolam identitas, Amazon Cognito membuat satu atau dua peran IAM (satu untuk identitas terotentikasi, dan satu untuk identitas “tamu” tidak terotentikasi) yang menentukan izin bagi pengguna Amazon Cognito.
3. Integrasikan AWS Amplify dengan aplikasi Anda, dan impor file yang diperlukan untuk menggunakan Amazon Cognito.
4. Buat instance penyedia kredenSIAL Amazon Cognito, meneruskan ID kumpulan identitas, Akun AWS nomor Anda, dan Nama Sumber Daya Amazon (ARN) dari peran yang Anda kaitkan dengan kumpulan identitas. Wizard Amazon Cognito di dalam Konsol Manajemen AWS menyediakan kode sampel untuk membantu Anda memulai.

5. Saat aplikasi mengakses AWS sumber daya, berikan instance penyedia kredenSIAL ke objek klien, yang meneruskan kredentif keamanan sementara ke klien. Izin untuk kredensial didasarkan pada peran atau peran-peran yang Anda tetapkan sebelumnya.

Untuk informasi selengkapnya, lihat berikut ini:

- Masuk (Android) di Dokumentasi Ker AWS Amplify angka Kerja.
- Masuk (iOS) di Dokumentasi Ker AWS Amplify angka Kerja.

Federasi OIDC untuk aplikasi seluler

Untuk hasil terbaik, gunakan Amazon Cognito sebagai broker identitas Anda untuk hampir semua skenario federasi OIDC. Amazon Cognito mudah digunakan dan menyediakan kemampuan tambahan seperti akses anonim (tidak terotentikasi), dan sinkronisasi data pengguna di seluruh perangkat dan penyedia. Namun, jika Anda telah membuat aplikasi yang menggunakan federasi OIDC dengan memanggil `AssumeRoleWithWebIdentity` API secara manual, Anda dapat terus menggunakannya dan aplikasi Anda akan tetap berfungsi dengan baik.

Proses untuk menggunakan federasi OIDC tanpa Amazon Cognito mengikuti garis besar umum ini:

1. Daftarkan diri Anda sebagai developer dengan penyedia identitas eksternal (IdP) dan konfigurasi aplikasi Anda dengan IdP, yang memberikan ID unik untuk aplikasi Anda. (Berbeda IdPs menggunakan terminologi yang berbeda untuk proses ini. Garis besar ini menggunakan istilah konfigurasi untuk proses mengidentifikasi aplikasi Anda dengan IdP.) Setiap IdP memberi Anda ID aplikasi yang unik untuk IdP tersebut, jadi jika Anda mengonfigurasi aplikasi yang sama dengan beberapa IdPs, aplikasi Anda akan memiliki beberapa ID aplikasi. Anda dapat mengonfigurasi beberapa aplikasi dengan setiap penyedia.

   Tautan eksternal berikut memberikan informasi tentang penggunaan beberapa penyedia identitas yang umum digunakan (IdPs):

   - Login dengan Amazon Developer Center
   - Tambahkan Login Facebook ke Aplikasi atau Website Anda di situs pengembang Facebook.
   - Menggunakan OAuth 2.0 untuk Login (OpenID Connect) di situs developer Google.

> **Important**
>
> Jika Anda menggunakan penyedia identitas OIDC dari Google, Facebook, atau Amazon Cognito, jangan membuat penyedia identitas IAM terpisah di. Konsol Manajemen AWS AWS memiliki penyedia identitas OIDC ini bawaan dan tersedia untuk Anda gunakan. Lewati langkah ini dan langsung buat peran baru menggunakan penyedia identitas Anda.

2. Jika Anda menggunakan IdP selain Google, Facebook, atau Amazon Cognito yang kompatibel dengan OIDC, buat entitas penyedia identitas IAM untuk itu.
3. Di dalam IAM, buat satu atau beberapa peran. Untuk setiap peran, tentukan siapa yang dapat mengambil peran (kebijakan kepercayaan) dan izin apa yang dimiliki pengguna aplikasi (kebijakan izin). Biasanya, Anda membuat satu peran untuk setiap IdP yang mendukung aplikasi. Misalnya, Anda dapat membuat peran yang diasumsikan oleh aplikasi jika pengguna masuk melalui Login dengan Amazon, peran kedua untuk aplikasi yang sama jika pengguna masuk melalui Facebook, dan peran ketiga untuk aplikasi jika pengguna masuk melalui Google. Untuk hubungan kepercayaan, tentukan IdP (like Amazon.com) sebagai `Principal` (entitas tepercaya), dan sertakan yang cocok dengan `Condition` ID aplikasi yang ditetapkan IdP. Contoh peran untuk penyedia yang berbeda dijelaskan dalam Membuat peran untuk penyedia identitas pihak ketiga.
4. Dalam aplikasi, autentikasi pengguna Anda dengan IdP. Spesifikasi tentang cara melakukan ini bervariasi sesuai dengan IdP yang Anda gunakan (Masuk dengan Amazon, Facebook, atau Google) dan di platform mana aplikasi Anda berjalan. Misalnya, metode otentikasi aplikasi Android dapat berbeda dari aplikasi iOS atau aplikasi JavaScript-based web.

   Biasanya, jika pengguna belum masuk, IdP menampilkan halaman masuk. Setelah IdP mengautentikasi pengguna, IdP mengembalikan informasi dengan token autentikasi tentang pengguna ke aplikasi Anda. Informasi yang disertakan tergantung pada apa yang diekspos IdP dan informasi apa yang ingin dibagikan oleh pengguna. Anda dapat menggunakan informasi ini di aplikasi Anda.
5. Di aplikasi Anda, buatlah panggilan berhenti menandatangani ke `AssumeRoleWithWebIdentity` tindakan untuk meminta kredensial keamanan sementara. Dalam permintaan, Anda meneruskan token otentikasi IdP dan menentukan Nama Sumber Daya Amazon (ARN) untuk peran IAM yang Anda buat untuk IdP tersebut. AWS memverifikasi bahwa token tepercaya dan valid, dan jika demikian, mengembalikan kredentif keamanan sementara ke aplikasi Anda yang memiliki izin untuk peran yang Anda beri nama dalam permintaan. Jawaban juga mencakup metadata tentang pengguna dari IdP, seperti ID pengguna unik yang dikaitkan IdP dengan pengguna.

6. Menggunakan kredentif keamanan sementara dari `AssumeRoleWithWebIdentity` respons, aplikasi Anda membuat permintaan yang ditandatangani ke operasi AWS API. Informasi ID pengguna dari IdP dapat membedakan pengguna di aplikasi Anda. Misalnya, Anda dapat memasukkan objek ke folder Amazon S3 yang menyertakan ID pengguna sebagai awalan atau akhiran. Hal ini memungkinkan Anda membuat kebijakan kontrol akses yang mengunci folder sehingga hanya pengguna dengan ID tersebut yang tidak dapat mengaksesnya. Untuk informasi selengkapnya, lihat AWS STS prinsipal pengguna federasi.
7. Aplikasi Anda harus menyimpan kredensial keamanan sementara sehingga Anda tidak perlu mendapatkan kredensial baru setiap kali aplikasi perlu membuat permintaan ke AWS. Secara default, kredensial akan berlaku selama satu jam. Saat kredensial kedaluwarsa (atau sebelum itu), Anda melakukan panggilan lain ke `AssumeRoleWithWebIdentity` untuk mendapatkan satu set kredensial keamanan sementara baru. Tergantung pada IdP dan cara mereka mengelola token, Anda mungkin harus menyegarkan token IdP sebelum melakukan panggilan baru ke`AssumeRoleWithWebIdentity` , karena token IdP juga biasanya kedaluwarsa setelah waktu tetap. Jika Anda menggunakan AWS SDK untuk iOS atau AWS SDK untuk Android, Anda dapat menggunakan AmazonSTSCredentialsProvider tindakan tersebut, yang mengelola kredenSIAL sementara IAM, termasuk menyegarkannya sesuai kebutuhan.

Federasi OIDC

Bayangkan Anda membuat aplikasi yang mengakses AWS sumber daya, seperti Tindakan yang menggunakan GitHub alur kerja untuk mengakses Amazon S3 dan DynamoDB.

Saat Anda menggunakan alur kerja ini, Anda membuat permintaan ke AWS layanan yang harus ditandatangani dengan kunci AWS akses. Namun, kami sangat menyarankan agar Anda tidak menyimpan AWS kredenSIAL jangka panjang di aplikasi di luar AWS. Sebagai gantinya, konfigurasikan aplikasi Anda untuk meminta kredenSIAL AWS keamanan sementara secara dinamis bila diperlukan menggunakan federasi OIDC. KredenSIAL sementara yang disediakan memet AWS akan ke peran yang hanya memiliki izin yang diperlukan untuk melakukan tugas yang diperlukan oleh aplikasi.

Dengan federasi OIDC, Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Sebagai gantinya, Anda dapat menggunakan OIDC dalam aplikasi, seperti Ac GitHub tions atau IdP lain yang kompatibel dengan OpenID Connect (OIDC), untuk diautentikasi. AWS Mereka menerima token otentikasi, yang dikenal sebagai Token Web JSON (JWT), dan kemudian menukar token itu dengan kredenSIAL keamanan sementara di peta AWS itu ke peran IAM dengan izin untuk menggunakan sumber daya tertentu di Anda. Akun AWS Menggunakan

IdP membantu Anda menjaga keamanan karena Akun AWS Anda tidak perlu menyematkan dan mendistribusikan kredentif keamanan jangka panjang dengan aplikasi Anda.

Federasi OIDC mendukung otentikasi mesin-ke-mesin (seperti CI/CD pipeline, skrip otomatis, dan aplikasi tanpa server) dan otentikasi pengguna manusia. Untuk skenario otentikasi pengguna manusia di mana Anda perlu mengelola pendaftaran pengguna, masuk, dan profil pengguna, pertimbangkan untuk menggunakan Amazon Cognito sebagai broker identitas. Untuk detail tentang menggunakan Amazon Cognito dengan OIDC, lihat. Amazon Cognito untuk aplikasi seluler

> **Note**
>
> Token Web JSON (JWT) yang dikeluarkan oleh penyedia identitas OpenID Connect (OIDC) berisi waktu kedaluwarsa dalam `exp` klaim yang menentukan kapan token kedaluwarsa. IAM menyediakan jendela lima menit di luar waktu kedaluwarsa yang ditentukan dalam JWT untuk memperhitungkan kemiringan jam, sebagaimana diizinkan oleh standar OpenID Connect (OIDC) Core 1.0. Ini berarti OIDC JWT yang diterima oleh IAM setelah waktu kedaluwarsa tetapi dalam jendela lima menit ini diterima untuk evaluasi dan pemrosesan lebih lanjut.

Topik

- Sumber daya tambahan untuk federasi OIDC
- Membuat penyedia identitas OpenID Connect (OIDC) di IAM
- Dapatkan sidik jari untuk penyedia identitas OpenID Connect
- Identity-provider kontrol untuk penyedia OIDC bersama

Sumber daya tambahan untuk federasi OIDC

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang federasi OIDC:

- Gunakan OpenID Connect dalam alur GitHub kerja Anda dengan Mengon figurasi OpenID Connect di Amazon Web Services
- Identitas Amazon Cognito di Panduan Amplify Libraries untuk Android dan Amazon Cognito Identity di Amplify Libraries for Swift Guide.
- Cara menggunakan ID eksternal saat memberikan akses ke AWS sumber daya Anda di Blog AWS Keamanan memberikan panduan untuk mengonfigurasi akses lintas akun dan federasi identitas eksternal dengan aman.

Membuat penyedia identitas OpenID Connect (OIDC) di IAM

Penyedia identitas IAM OIDC adalah entitas di IAM yang menggambarkan layanan penyedia identitas eksternal (IdP) yang mendukung standar OpenID Connect (OIDC), seperti Google atau Salesforce. Anda menggunakan penyedia identitas IAM OIDC ketika Anda ingin membangun kepercayaan antara OIDC-compatible IdP dan Anda. Akun AWS Ini berguna saat membuat aplikasi seluler atau aplikasi web yang memerlukan akses ke AWS sumber daya, tetapi Anda tidak ingin membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Untuk informasi selengkapnya tentang skenario ini, lihat the section called “Federasi OIDC”.

Anda dapat membuat dan mengelola penyedia identitas IAM OIDC menggunakan Konsol Manajemen AWS,, Tools for Windows PowerShell, atau IAM API. AWS Command Line Interface

Setelah Anda membuat penyedia identitas IAM OIDC, Anda harus membuat satu atau lebih peran IAM. Peran adalah identitas AWS yang tidak memiliki kredensialnya sendiri (seperti yang dilakukan pengguna). Namun dalam konteks ini, peran secara dinamis ditetapkan ke prinsipal federasi OIDC yang diautentikasi oleh IdP organisasi Anda. Peran mengizinkan IdP organisasi Anda meminta kredensial keamanan sementara untuk akses ke AWS. Kebijakan yang ditetapkan untuk peran menentukan apa yang diizinkan untuk dilakukan pengguna AWS. Untuk membuat peran bagi penyedia identitas pihak ketiga, lihat Membuat peran untuk penyedia identitas pihak ketiga.

> **Important**
>
> Saat Anda mengonfigurasi kebijakan berbasis identitas untuk tindakan yang mendukung `oidc-provider` sumber daya, IAM mengevaluasi URL penyedia identitas OIDC lengkap, termasuk jalur yang ditentukan. Jika URL penyedia identitas OIDC Anda memiliki jalur, Anda harus menyertakan jalur tersebut di `oidc-provider` ARN sebagai nilai `Resource` elemen. Anda juga memiliki opsi untuk menambahkan garis miring maju dan wildcard (`/*`) ke domain URL atau menggunakan karakter wildcard (*dan?) di setiap titik di jalur URL. Jika URL penyedia identitas OIDC dalam permintaan tidak cocok dengan nilai yang ditetapkan di `Resource` elemen kebijakan, permintaan gagal.

Untuk memecahkan masalah umum dengan federasi IAM OIDC, lihat Menyel esaikan kesalahan yang terkait dengan OIDC di re:POST. AWS

Topik

- Prasyarat: Validasi konfigurasi penyedia identitas Anda

- Membuat dan mengelola penyedia OIDC (konsol)
- Membuat dan mengelola penyedia identitas IAM OIDC (AWS CLI)
- Membuat dan mengelola Penyedia Identitas OIDC (AWS API)

Prasyarat: Validasi konfigurasi penyedia identitas Anda

Sebelum Anda dapat membuat penyedia identitas IAM OIDC, Anda harus memiliki informasi berikut dari IdP Anda. Untuk informasi selengkapnya tentang mendapatkan Informasi konfigurasi penyedia OIDC, lihat dokumentasi untuk IdP Anda.

1. Tentukan URL penyedia identitas OIDC yang tersedia untuk umum. URL harus dimulai dengan S https://. esuai standar OIDC, komponen jalur diizinkan tetapi parameter kueri tidak. Biasanya, URL hanya terdiri dari nama host, seperti https://server.example.org atau https://example.com. URL tidak boleh berisi nomor port.
2. Tambahkan /.well- known/openid -configuration ke akhir URL penyedia identitas OIDC Anda untuk melihat dokumen konfigurasi dan metadata penyedia yang tersedia untuk umum. Anda harus memiliki dokumen penemuan dalam format JSON dengan dokumen konfigurasi penyedia dan metadata yang dapat diambil dari URL titik akhir penemuan penyedia OpenID Connect.
3. Konfirmasikan nilai-nilai berikut disertakan dalam informasi konfigurasi penyedia Anda. Jika konfigurasi openid Anda tidak memiliki salah satu bidang ini, Anda harus memperbarui dokumen penemuan Anda. Proses ini dapat bervariasi berdasarkan penyedia identitas Anda, jadi ikuti dokumentasi IdP Anda untuk menyelesaikan tugas ini.

    - penerbit: URL untuk domain Anda.
    - jwks_uri: Titik akhir JSON Web Key Set (JWKS) tempat IAM mendapatkan kunci publik Anda. Penyedia identitas Anda harus menyertakan titik akhir JSON Web Key Set (JWKS) dalam konfigurasi openid. URI ini menentukan tempat mendapatkan kunci publik yang digunakan untuk memverifikasi token yang ditandatangani dari penyedia identitas Anda.

    > **Note**
    >
    > JSON Web Key Set (JWKS) harus berisi setidaknya satu kunci dan dapat memiliki maksimum 100 kunci RSA dan 100 kunci EC. Jika JWKS penyedia identitas OIDC Anda berisi lebih dari 100 kunci RSA atau 100 kunci EC, `InvalidIdentityToken` pengecualian akan dikembalikan saat menggunakan operasi AssumeRoleWithWebIdentity API dengan JWT yang ditandatangani oleh jenis kunci yang melebihi batas 100 kunci. Misalnya, jika JWT ditandatangani

dengan algoritma RSA dan ada lebih dari 100 kunci RSA di JWKS penyedia Anda, pengecualian akan dikembalikan. `InvalidIdentityToken`

- claims_supported: Informasi tentang pengguna yang membantu Anda memastikan respons otentikasi OIDC dari IdP berisi atribut AWS yang diperlukan yang digunakan dalam kebijakan IAM untuk memeriksa izin untuk prinsipal federasi OIDC. Untuk daftar kunci kondisi IAM yang dapat digunakan untuk klaim, lihatKunci yang tersedia untuk AWS Federasi OIDC.
  - aud: Anda harus menentukan nilai klaim audiens yang dikeluarkan IdP Anda di Token Web JSON (JWT). Klaim audiens (aud) bersifat spesifik aplikasi dan mengidentifikasi penerima token yang dituju. Ketika Anda mendaftarkan aplikasi seluler atau web dengan penyedia OpenID Connect, mereka membuat ID klien yang mengidentifikasi aplikasi. ID klien adalah pengenal unik untuk aplikasi Anda yang diteruskan dalam klaim aud untuk otentikasi. Klaim aud harus sesuai dengan nilai Audiens saat membuat penyedia identitas IAM OIDC Anda.
  - iat: Klaim harus menyertakan nilai `iat` yang mewakili waktu token ID dikeluarkan.
  - iss: URL penyedia identitas. URL harus dimulai dengan https://dan harus sesuai dengan URL Penyedia yang disediakan untuk IAM. Sesuai standar OIDC, komponen jalur diizinkan tetapi parameter kueri tidak. Biasanya, URL hanya terdiri dari nama host, seperti https://server.example.org atau https://example.com. URL tidak boleh berisi nomor port.
- response_types_didukung: id_token
- subject_types_supported: publik
- id_token_signing_alg_values_didukung: RS256, RS384, RS512, ES256, ES384, ES512

> **Note**
>
> Anda dapat menyertakan klaim tambahan seperti `my_custom_claim` pada contoh di bawah ini; namun, AWS STS akan mengabaikan klaim tersebut.

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "my_custom_claim"
```

```
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256",
    "RS384",
    "RS512",
    "ES256",
    "ES384",
    "ES512"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Membuat dan mengelola penyedia OIDC (konsol)

Ikuti petunjuk ini untuk membuat dan mengelola penyedia identitas IAM OIDC di. Konsol Manajemen AWS

> **Important**
>
> Jika Anda menggunakan penyedia identitas OIDC dari Google, Facebook, atau Amazon Cognito, jangan membuat penyedia identitas IAM terpisah menggunakan prosedur ini. Penyedia identitas OIDC ini sudah terpasang AWS dan tersedia untuk Anda gunakan. Alih-alih, ikuti langkah-langkah untuk membuat peran baru bagi penyedia identitas Anda, lihat Membuat peran untuk federasi OpenID Connect (konsol).

Untuk membuat penyedia identitas IAM OIDC (konsol)

1. Sebelum Anda membuat penyedia identitas IAM OIDC, Anda harus mendaftarkan aplikasi Anda dengan IdP untuk menerima ID klien. ID klien (juga dikenal sebagai audiens) adalah pengenal unik untuk aplikasi yang diberikan kepada Anda saat mendaftarkan aplikasi dengan IdP. Untuk informasi lebih lanjut tentang mendapatkan ID klien, lihat dokumentasi untuk IdP.

> **Note**
>
> AWS mengamankan komunikasi dengan penyedia identitas OIDC (IdPs) menggunakan pustaka otoritas sertifikat root tepercaya (CA) kami untuk memverifikasi sertifikat TLS titik akhir JSON Web Key Set (JWKS). Jika IDP OIDC Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu CA tepercaya ini, hanya kemudian kami mengamankan komunikasi menggunakan sidik jari yang ditetapkan dalam konfigurasi IdP. AWS akan kembali ke verifikasi sidik jari jika kami tidak dapat mengambil sertifikat TLS atau jika TLS v1.3 diperlukan.

2. Buka konsol IAM di https://console.aws.amazon.com/iam/.
3. Di panel navigasi, pilih Penyedia identitas, lalu pilih Tambahkan penyedia.
4. Untuk Konfigurasi penyedia, pilih OpenID Connect.
5. Untuk URL Penyedia, ketik URL IdP. URL harus mematuhi pembatasan ini:
   - URL peka terhadap huruf besar-kecil.
   - URL harus dimulai dengan **`https://`**.
   - URL tidak boleh berisi nomor port.
   - Di dalam Anda Akun AWS, setiap penyedia identitas IAM OIDC harus menggunakan URL unik. Jika Anda mencoba mengirimkan URL yang telah digunakan untuk penyedia OpenID Connect di Akun AWS, Anda akan mendapatkan kesalahan.
6. Untuk Audiens, ketik ID klien aplikasi yang Anda daftarkan dengan IdP dan diterimaStep 1, dan yang membuat AWS permintaan. Jika memiliki ID klien tambahan (juga dikenal sebagai audiens) untuk IdP ini, Anda dapat menambahkannya nanti di halaman detail penyedia.

   > **Note**
   >
   > Jika token JWT IdP Anda menyertakan `azp` klaim, masukkan nilai ini sebagai nilai Audiens.
   > Jika penyedia identitas OIDC Anda menyetel keduanya `aud` dan `azp` klaim di token, AWS STS akan menggunakan nilai dalam `azp` klaim sebagai `aud` klaim.

7. (Opsional) Untuk Tambah tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS . Untuk mempelajari selengkapnya tentang menandai

penyedia identitas IAM OIDC, lihat. Menandai penyedia identitas OpenID Connect (OIDC) Pilih Tambahkan tanda. Masukkan nilai untuk setiap pasangan nilai-kunci tanda.

8. Verifikasi informasi yang telah Anda berikan. Setelah selesai, pilih Tambahkan penyedia. IAM akan mencoba mengambil dan menggunakan sidik jari CA perantara teratas dari sertifikat server IDC IdP untuk membuat penyedia identitas IAM OIDC.

> **Note**
>
> Rantaian sertifikat penyedia identitas OIDC harus dimulai dengan domain atau URL penerbit, kemudian sertifikat perantara, dan diakhiri dengan sertifikat root. Jika urutan rantai sertifikat berbeda atau menyertakan sertifikat duplikat atau tambahan, maka Anda menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi Token Web JSON (JWT). Perbaiki urutan sertifikat dalam rantai yang dikembalikan dari server untuk menyelesaikan kesalahan. Untuk informasi selengkapnya tentang standar rantai sertifikat, lihat certificate_list di RFC 5246 di situs web Seri RFC.

9. Tetapkan peran IAM ke penyedia identitas Anda untuk memberikan identitas pengguna eksternal yang dikelola oleh penyedia identitas Anda izin untuk mengakses AWS sumber daya di akun Anda. Untuk mempelajari lebih lanjut tentang menciptakan peran untuk federasi identitas, lihat Membuat peran untuk penyedia identitas pihak ketiga.

> **Note**
>
> OIDC yang IdPs digunakan dalam kebijakan kepercayaan peran harus berada di akun yang sama dengan peran yang mempercayainya.

Untuk menambahkan atau menghapus sidik jari untuk penyedia identitas IAM OIDC (konsol)

> **Note**
>
> AWS mengamankan komunikasi dengan penyedia identitas OIDC (IdPs) menggunakan pustaka otoritas sertifikat root tepercaya (CA) kami untuk memverifikasi sertifikat TLS titik akhir JSON Web Key Set (JWKS). Jika IDP OIDC Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu CA tepercaya ini, hanya kemudian kami mengamankan komunikasi menggunakan sidik jari yang ditetapkan dalam konfigurasi IdP. AWS akan

kembali ke verifikasi sidik jari jika kami tidak dapat mengambil sertifikat TLS atau jika TLS v1.3 diperlukan.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Penyedia identitas. Kemudian pilih nama penyedia identitas IAM yang ingin Anda perbarui.
3. Pilih tab Veri fikasi Endpoint, lalu di bagian T humbprints, pilih Kel ola. Untuk memasukkan nilai sidik jari baru, pilih Tambahkan sidik jari. Untuk menghapus sidik jari, pilih Hapus di samping sidik jari yang ingin Anda hapus.

   > **Note**
   >
   > Seorang penyedia identitas IAM OIDC harus memiliki setidaknya satu dan dapat memiliki maksimal lima sidik jari.

   Setelah selesai, pilih Simpan perubahan.

Untuk menambahkan audiens untuk penyedia identitas IAM OIDC (konsol)

1. Di panel navigasi, pilih Penyedia identitas, kemudian pilih nama penyedia identitas IAM yang ingin Anda perbarui.
2. Di bagian Audiens, pilih Tindakan dan pilih Tambahkan audiens.
3. Ketik ID klien aplikasi yang Anda daftarkan dengan IdP dan terimaStep 1, dan itu akan membuat AWS permintaan. Lalu, pilih Tambahkan audiens.

   > **Note**
   >
   > Seorang penyedia identitas IAM OIDC harus memiliki setidaknya satu dan dapat memiliki maksimal 100 audiens.

Untuk menghapus audiens untuk penyedia identitas IAM OIDC (konsol)

1. Di panel navigasi, pilih Penyedia identitas, kemudian pilih nama penyedia identitas IAM yang ingin Anda perbarui.

2. Di bagian Audiens, pilih tombol radio di samping audiens yang ingin Anda hapus, lalu pilih Tindakan.
3. Pilih Hapus audiens. Jendela baru terbuka.
4. Jika Anda menghapus audiens, identitas yang bergabung dengan audiens tidak dapat mengambil peran yang terkait dengan audiens. Di jendela, baca peringatan dan konfirmasikan bahwa Anda ingin menghapus audiens dengan mengetikkan kata `remove` di bidangnya.
5. Pilih Hapus untuk menghapus audiens.

Untuk menghapus penyedia identitas IAM OIDC (konsol)

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, silakan pilih Penyedia identitas.
3. Pilih kotak centang di sebelah penyedia identitas IAM yang ingin Anda hapus. Jendela baru terbuka.
4. Konfirmasikan bahwa Anda ingin menghapus penyedia dengan mengetik kata `delete` di bidangnya. Lalu, pilih Hapus.

Membuat dan mengelola penyedia identitas IAM OIDC (AWS CLI)

Anda dapat menggunakan AWS CLI perintah berikut untuk membuat dan mengelola penyedia identitas IAM OIDC.

Untuk membuat penyedia identitas IAM OIDC (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, jalankan perintah berikut:
   - `aws iam list-open-id-connect-providers`
2. Untuk membuat penyedia identitas OIDC IAM baru, jalankan perintah berikut:
   - `aws iam create-open-id-connect-provider`

Untuk memperbarui daftar sidik jari sertifikat server untuk penyedia identitas IAM OIDC yang ada (AWS CLI)

- Untuk memperbarui daftar sidik jari sertifikat server untuk penyedia identitas IAM OIDC, jalankan perintah berikut:

- `aws iam update-open-id-connect-provider-thumbprint`

Untuk menandai penyedia identitas IAM OIDC yang ada (AWS CLI)

- Untuk menandai penyedia identitas IAM OIDC yang ada, jalankan perintah berikut:
  - `aws iam tag-open-id-connect-provider`

Untuk mencantumkan tag untuk penyedia identitas IAM OIDC yang ada (AWS CLI)

- Untuk mencantumkan tag untuk penyedia identitas IAM OIDC yang ada, jalankan perintah berikut:
  - `aws iam list-open-id-connect-provider-tags`

Untuk menghapus tag pada penyedia identitas IAM OIDC (AWS CLI)

- Untuk menghapus tag pada penyedia identitas IAM OIDC yang ada, jalankan perintah berikut:
  - `aws iam untag-open-id-connect-provider`

Untuk menambah atau menghapus ID klien dari penyedia identitas IAM OIDC yang ada (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, jalankan perintah berikut:
   - `aws iam list-open-id-connect-providers`
2. (Opsional) Untuk mendapatkan informasi terperinci tentang penyedia identitas IAM OIDC, jalankan perintah berikut:
   - `aws iam get-open-id-connect-provider`
3. Untuk menambahkan ID klien baru ke penyedia identitas IAM OIDC yang ada, jalankan perintah berikut:
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Untuk menghapus klien dari penyedia identitas IAM OIDC yang ada, jalankan perintah berikut:

- `aws iam remove-client-id-from-open-id-connect-provider`

Untuk menghapus penyedia identitas IAM OIDC (AWS CLI)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, jalankan perintah berikut:
   - `aws iam list-open-id-connect-providers`
2. (Opsional) Untuk mendapatkan informasi terperinci tentang penyedia identitas IAM OIDC, jalankan perintah berikut:
   - `aws iam get-open-id-connect-provider`
3. Untuk menghapus penyedia identitas IAM OIDC, jalankan perintah berikut:
   - `aws iam delete-open-id-connect-provider`

Membuat dan mengelola Penyedia Identitas OIDC (AWS API)

Anda dapat menggunakan perintah IAM API berikut untuk membuat dan mengelola penyedia OIDC.

Untuk membuat penyedia identitas IAM OIDC (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, panggil operasi berikut:
   - `ListOpenIDConnectProviders`
2. Untuk membuat penyedia identitas IAM OIDC baru, panggil operasi berikut:
   - `CreateOpenIDConnectProvider`

Untuk memperbarui daftar sidik jari sertifikat server untuk penyedia identitas IAM OIDC yang ada (AWS API)

- Untuk memperbarui daftar sidik jari sertifikat server untuk penyedia identitas IAM OIDC, panggil operasi berikut:
  - `UpdateOpenIDConnectProviderThumbprint`

Untuk menandai penyedia identitas IAM OIDC yang ada (AWS API)

- Untuk menandai penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
  - `TagOpenIDConnectProvider`

Untuk mencantumkan tag untuk penyedia identitas IAM OIDC yang ada (AWS API)

- Untuk mencantumkan tag untuk penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
  - `ListOpenIDConnectProviderTags`

Untuk menghapus tag pada penyedia identitas IAM OIDC yang ada (AWS API)

- Untuk menghapus tag pada penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
  - `UntagOpenIDConnectProvider`

Untuk menambah atau menghapus ID klien dari penyedia identitas IAM OIDC yang ada (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, panggil operasi berikut:
   - `ListOpenIDConnectProviders`
2. (Opsional) Untuk mendapatkan informasi terperinci tentang penyedia identitas IAM OIDC, panggil operasi berikut:
   - `GetOpenIDConnectProvider`
3. Untuk menambahkan ID klien baru ke penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
   - `AddClientIDToOpenIDConnectProvider`
4. Untuk menghapus ID klien ke penyedia identitas IAM OIDC yang ada, panggil operasi berikut:
   - `RemoveClientIDFromOpenIDConnectProvider`

Untuk menghapus penyedia identitas IAM OIDC (AWS API)

1. (Opsional) Untuk mendapatkan daftar semua penyedia identitas IAM OIDC di AWS akun Anda, panggil operasi berikut:
   - `ListOpenIDConnectProviders`
2. (Opsional) Untuk mendapatkan informasi terperinci tentang penyedia identitas IAM OIDC, panggil operasi berikut:
   - `GetOpenIDConnectProvider`
3. Untuk menghapus penyedia identitas IAM OIDC, hubungi operasi berikut:
   - `DeleteOpenIDConnectProvider`

Dapatkan sidik jari untuk penyedia identitas OpenID Connect

Saat Anda membuat penyedia identitas OpenID Connect (OIDC) di IAM, IAM memerlukan sidik jari untuk otoritas sertifikat perantara teratas (CA) yang menandatangani sertifikat yang digunakan oleh penyedia identitas eksternal (IdP). Sidik jari adalah tanda tangan untuk sertifikat CA yang digunakan untuk mengeluarkan sertifikat untuk OIDC-compatible IdP. Saat Anda membuat penyedia identitas IAM OIDC, Anda mempercayai identitas yang diautentikasi oleh IdP tersebut untuk memiliki akses ke identitas Anda. Akun AWS Dengan menggunakan sidik jari sertifikat CA, Anda mempercayai sertifikat apa pun yang dikeluarkan oleh CA tersebut dengan nama DNS yang sama dengan yang terdaftar. Ini menghilangkan perlunya memperbarui kepercayaan di setiap akun ketika Anda memperpanjang sertifikat tanda tangan IdP.

> **Important**
>
> Dalam kebanyakan kasus, server federasi menggunakan dua sertifikat yang berbeda:
>
> - Yang pertama membuat koneksi HTTPS antara AWS dan IdP Anda. Ini harus dikeluarkan oleh CA root publik yang terkenal, seperti AWS Certificate Manager. Hal ini memungkinkan klien untuk memeriksa keandalan dan status sertifikat.
> - Yang kedua digunakan untuk mengenkripsi token, dan harus ditandatangani oleh CA root pribadi atau publik.

Anda dapat membuat penyedia identitas IAM OIDC dengan AWS Command Line Interface, Tools for Windows PowerShell, atau IAM API. Saat Anda menggunakan metode ini, Anda memiliki opsi untuk memberikan sidik jari secara manual. Jika Anda memilih untuk tidak menyertakan sidik jari, IAM akan mengambil sidik jari CA perantara teratas dari sertifikat server IDC IdP. Jika Anda memilih untuk menyertakan sidik jari, Anda harus mendapatkan sidik jari secara manual dan menyediakannya. AWS

Saat Anda membuat penyedia identitas OIDC dengan konsol IAM, IAM mencoba mengambil sidik jari CA perantara teratas dari sertifikat server IDC IdP untuk Anda.

Sebaiknya Anda juga mendapatkan sidik jari untuk IDC IdP Anda secara manual dan memverifikasi bahwa IAM mengambil sidik jari yang benar. Untuk informasi selengkapnya tentang mendapatkan sidik jari sertifikat, lihat bagian berikut.

> **Note**
>
> AWS mengamankan komunikasi dengan penyedia identitas OIDC (IdPs) menggunakan pustaka otoritas sertifikat root tepercaya (CA) kami untuk memverifikasi sertifikat TLS titik akhir JSON Web Key Set (JWKS). Jika IDP OIDC Anda bergantung pada sertifikat yang tidak ditandatangani oleh salah satu CA tepercaya ini, hanya kemudian kami mengamankan komunikasi menggunakan sidik jari yang ditetapkan dalam konfigurasi IdP. AWS akan kembali ke verifikasi sidik jari jika kami tidak dapat mengambil sertifikat TLS atau jika TLS v1.3 diperlukan.

> **Important**
>
> Jika penyedia identitas OIDC menggunakan host yang berbeda untuk titik akhir penemuan (URL penerbit) dan titik akhir JWKS (`jwks_uri`), Anda harus menyertakan sidik jari untuk kedua titik akhir dalam daftar sidik jari penyedia OIDC Anda. AWS memvalidasi sidik jari dari kedua titik akhir saat kembali ke verifikasi sidik jari. Jika salah satu sidik jari hilang dari daftar yang dikonfigurasi, otentikasi gagal.
>
> Sebagian besar penyedia OIDC meng-host kedua titik akhir pada domain yang sama, sehingga sidik jari tunggal mencakup keduanya. Anda hanya perlu menambahkan sidik jari terpisah jika penyedia Anda menggunakan host atau sertifikat yang berbeda untuk dua titik akhir.

Dapatkan sidik jari sertifikat

Anda menggunakan browser web dan alat baris perintah OpenSSL untuk mendapatkan sidik jari sertifikat untuk penyedia OIDC. Namun, Anda tidak perlu mendapatkan sidik jari sertifikat secara manual untuk membuat penyedia identitas IAM OIDC. Anda dapat menggunakan prosedur berikut untuk mendapatkan sidik jari sertifikat penyedia OIDC Anda.

Untuk mendapatkan sidik jari untuk IdP OIDC

1. Sebelum mendapatkan sidik jari untuk IdP OIDC, Anda harus mendapatkan alat baris perintah OpenSSL. Anda menggunakan alat ini untuk mengunduh rantai sertifikat IdP OIDC dan menghasilkan sidik jari untuk sertifikat akhir dalam rantai sertifikat. Jika Anda perlu menginstal dan mengonfigurasi OpenSSL, ikuti petunjuk di Instal OpenSSL dan Konfigurasi OpenSSL.
2. Mulailah dengan URL IdP OIDC (misalnya, `https://server.example.com`), lalu tambahkan `/.well-known/openid-configuration` untuk membentuk URL bagi dokumen konfigurasi IdP, seperti berikut:

   `https://server.example.com/.well-known/openid-configuration`

   Buka URL ini di browser web, ganti *`server.example.com`* dengan nama server IdP Anda.
3. Pada dokumen yang ditampilkan, gunakan fitur Cari pada browser web untuk menemukan teks `"jwks_uri"`. Setelah teks `"jwks_uri"`, terdapat tanda titik dua (:) diikuti oleh URL. Salin nama domain yang sepenuhnya memenuhi syarat dari URL. Jangan sertakan `https://` atau jalur apa pun yang mengikuti domain tingkat atas.

   ```
   {
    "issuer": "https://accounts.example.com",
    "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
    "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
    "token_endpoint": "https://oauth2.exampleapis.com/token",
    "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
    "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
    "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
   ...
   ```

> **Note**
>
> Jika penemuan dan titik akhir JWKS Anda menggunakan host yang berbeda, catat nama domain yang sepenuhnya memenuhi syarat dari URL penerbit Anda (URL yang Anda gunakanStep 2, tanpa jalur). `/.well-known/openid-configuration` Jika domain ini berbeda dari `jwks_uri` domain yang Anda salin pada langkah ini, lakukan Step 6 untuk Step 4 kedua domain dan sertakan kedua sidik jari saat membuat penyedia OIDC Anda.

4. Gunakan alat baris perintah OpenSSL untuk menjalankan perintah berikut. Ganti *keys.example.com* dengan nama domain yang Anda dapatkanStep 3.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. Di jendela perintah Anda, gulir ke atas hingga Anda melihat sertifikat yang serupa dengan contoh berikut. Jika Anda melihat lebih dari satu sertifikat, cari sertifikat terakhir yang ditampilkan (di bagian akhir output perintah). Ini berisi sertifikat CA perantara teratas dalam rantai otoritas sertifikat.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
 -----END CERTIFICATE-----
```

Menyalin sertifikat (termasuk baris `-----BEGIN CERTIFICATE-----` dan `-----END CERTIFICATE-----`) dan tempel ke dalam file teks. Lalu simpan file dengan nama file **`certificate.crt`**.

> **Note**
>
> Rantaian sertifikat penyedia identitas OIDC harus dimulai dengan domain atau URL penerbit, menyertakan sertifikat perantara (jika ada), dan diakhiri dengan sertifikat root. Jika urutan rantai sertifikat berbeda atau menyertakan sertifikat duplikat atau tambahan, Anda akan menerima kesalahan ketidakcocokan tanda tangan dan STS gagal memvalidasi Token Web JSON (JWT). Perbaiki urutan sertifikat dalam rantai yang dikembalikan dari server untuk menyelesaikan kesalahan. Untuk informasi selengkapnya tentang standar rantai sertifikat, lihat certificate_list di RFC 5246 di situs web Seri RFC.

6. Gunakan alat baris perintah OpenSSL untuk menjalankan perintah berikut.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Jendela perintah Anda menampilkan sidik jari sertifikat, yang terlihat mirip dengan contoh berikut:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Hapus karakter titik dua (:) dari string ini untuk membuat sidik jari akhir, seperti ini:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

> **Note**
>
> Jika titik akhir penemuan Anda menggunakan host yang berbeda dari titik akhir JWKS Anda, ulangi Step 4 melalui Step 6 menggunakan domain titik akhir penemuan (domain URL penerbit Anda). Sertakan kedua sidik jari saat membuat atau memperbarui penyedia identitas OIDC Anda menggunakan CreateOpenIDConnectProvider atau. UpdateOpenIDConnectProviderThumbprint

7. Jika Anda membuat penyedia identitas IAM OIDC dengan AWS CLI, Tools for Windows, atau IAM API PowerShell, memberikan sidik jari adalah opsional. Jika Anda memilih untuk

tidak menyertakan sidik jari selama pembuatan, IAM akan mengambil sidik jari CA perantara teratas dari sertifikat server IDC IdP. Setelah penyedia identitas IAM OIDC dibuat, Anda dapat membandingkan sidik jari ini dengan sidik jari yang diambil oleh IAM.

Jika Anda membuat penyedia identitas IAM OIDC di konsol IAM, konsol akan mencoba mengambil sidik jari CA perantara teratas dari sertifikat server IDC IdP untuk Anda. Anda dapat membandingkan sidik jari ini dengan sidik jari yang diambil oleh IAM. Setelah penyedia identitas IAM OIDC dibuat, Anda dapat melihat sidik jari untuk penyedia identitas IAM OIDC di tab Verifikasi titik akhir pada halaman konsol Ringkasan penyedia OIDC.

> **Important**
>
> Jika sidik jari yang Anda peroleh tidak cocok dengan yang Anda lihat di detail sidik jari penyedia identitas IAM OIDC, Anda tidak boleh menggunakan penyedia OIDC. Sebagai gantinya, Anda harus menghapus penyedia OIDC yang dibuat dan kemudian mencoba lagi untuk membuat penyedia OIDC setelah beberapa waktu berlalu. Pastikan sidik jari cocok sebelum Anda menggunakan penyedia. Jika sidik jari masih belum sama setelah percobaan kedua, gunakan Forum IAM untuk menghubungi AWS.

Instal OpenSSL

Jika Anda belum menginstal OpenSSL, ikuti petunjuk di bagian ini.

Untuk menginstal OpenSSL di Linux atau Unix

1. Buka OpenSSL: Sumber, Tarballs ()https://openssl.org/source/.
2. Unduh sumber terbaru dan buat paketnya.

Untuk menginstal OpenSSL di Windows

1. Buka OpenSSL: Binary Distribu https://wiki.openssl.org/index.php/Binaries tions () untuk daftar situs tempat Anda dapat menginstal versi Windows.
2. Ikuti petunjuk di situs yang Anda pilih untuk memulai pemasangan.
3. Jika Anda diminta memasang Microsoft Visual C++ 2008 Redistributables dan itu belum diinstal di sistem Anda, pilih tautan unduhan yang sesuai dengan lingkungan Anda. Ikuti petunjuk yang diberikan oleh Microsoft Visual C++ 2008 Redistributable Setup Wizard.

> **Note**
>
> Jika Anda tidak yakin apakah Microsoft Visual C++ 2008 Redistributables sudah diinstal pada sistem Anda, Anda dapat mencoba menginstal OpenSSL terlebih dahulu. Penginstal OpenSSL menampilkan peringatan jika Microsoft Visual C++ 2008 Redistributables belum diinstal. Pastikan Anda menginstal arsitektur (32-bit atau 64-bit) yang cocok dengan versi OpenSSL yang Anda instal.

4. Setelah Anda menginstal Microsoft Visual C++ 2008 Redistributables, pilih versi binari OpenSSL yang sesuai untuk lingkungan Anda dan simpan file secara lokal. Mulai OpenSSL Setup Wizard.
5. Ikuti petunjuk yang dijelaskan dalam OpenSSL Setup Wizard.

Konfigurasi OpenSSL

Sebelum Anda menggunakan perintah OpenSSL, Anda harus mengkonfigurasi sistem operasi sehingga memiliki informasi tentang lokasi di mana OpenSSL diinstal.

Untuk mengkonfigurasi OpenSSL di Linux atau Unix

1. Pada baris perintah, atur `OpenSSL_HOME` variabel ke lokasi instalasi OpenSSL:

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. Atur jalur untuk menyertakan instalasi OpenSSL:

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

> **Note**
>
> Perubahan apa pun yang Anda lakukan terhadap variabel dengan perintah `export` hanya berlaku untuk sesi saat ini. Anda dapat membuat perubahan terus-menerus pada variabel lingkungan dengan menyetelnya di file konfigurasi shell Anda. Untuk informasi lebih lanjut, lihat dokumentasi untuk sistem operasi Anda.

Untuk mengkonfigurasi OpenSSL pada Windows

1. Buka jendela Prompt Perintah.
2. Atur `OpenSSL_HOME` variabel ke lokasi instalasi OpenSSL:

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Atur `OpenSSL_CONF` variabel ke lokasi file konfigurasi dalam instalasi OpenSSL Anda:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Atur jalur untuk menyertakan instalasi OpenSSL:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Perubahan apa pun yang Anda lakukan terhadap variabel lingkungan Windows dalam jendela Prompt Perintah hanya valid untuk sesi baris perintah saat ini. Anda dapat membuat perubahan persisten pada variabel lingkungan dengan mengaturnya sebagai properti sistem. Prosedur yang tepat tergantung pada versi Windows yang Anda gunakan. (Misalnya, di Windows 7, buka Panel Kontrol, Sistem dan Keamanan, Sistem. Kemudian pilih Peng aturan sistem lanjutan, tab L anjutan, Variabel Lingkungan.) Untuk informasi selengkapnya, lihat dokumentasi Windows.

Identity-provider kontrol untuk penyedia OIDC bersama

Untuk penyedia identitas OpenID Connect (OIDC) bersama yang diakui (IdPs), IAM memerlukan evaluasi eksplisit atas klaim spesifik dalam kebijakan kepercayaan peran. Klaim yang diperlukan ini, yang disebut kontrol penyedia identitas, dievaluasi oleh IAM selama pembuatan peran dan pembaruan kebijakan kepercayaan. Jika kebijakan kepercayaan peran tidak mengevaluasi kontrol yang diperlukan oleh iDP OIDC bersama, pembuatan atau pembaruan peran akan gagal. Ini memastikan bahwa hanya identitas resmi dari organisasi yang dituju yang dapat mengambil peran dan mengakses AWS sumber daya. Kontrol keamanan ini sangat penting ketika penyedia OIDC dibagi di beberapa AWS pelanggan.

Identity-provider kontrol tidak akan dievaluasi oleh IAM untuk kebijakan kepercayaan peran OIDC yang ada. Untuk setiap modifikasi kebijakan kepercayaan peran untuk peran OIDC yang ada, IAM akan mengharuskan kontrol penyedia identitas dimasukkan dalam kebijakan kepercayaan peran.

Jenis penyedia OIDC

IAM mengkategorikan penyedia identitas OIDC menjadi dua jenis berbeda: pribadi dan bersama. IdP OIDC pribadi dapat dimiliki dan dikelola oleh satu organisasi atau dapat menjadi penyewa penyedia SaaS, dengan URL Penerbit OIDC-nya berfungsi sebagai pengidentifikasi unik khusus untuk organisasi itu. Sebaliknya, iDP OIDC bersama digunakan di beberapa organisasi, di mana URL Penerbit OIDC mungkin identik untuk semua organisasi yang menggunakan penyedia identitas bersama tersebut.

Tabel di bawah ini menguraikan perbedaan utama antara penyedia OIDC pribadi dan bersama:

Karakteristik	Penyedia OIDC Pribadi	Penyedia OIDC Bersama
Penerbit	Unik untuk organisasi	Dibagikan di berbagai organisasi
Informasi Sewa	Dikomunikasikan melalui Emiten unik	Dikomunikasikan melalui klaim di JWT
Persyaratan Kebijakan Kepercayaan	Tidak diperlukan evaluasi klaim khusus	Evaluasi klaim spesifik yang diperlukan

Penyedia identitas OIDC bersama dengan kontrol penyedia identitas

Saat Anda membuat atau memodifikasi penyedia OIDC di IAM, sistem secara otomatis mengidentifikasi dan mengevaluasi klaim yang diperlukan untuk penyedia OIDC bersama yang diakui. Jika kontrol penyedia identitas tidak dikonfigurasi dalam kebijakan kepercayaan peran, pembuatan atau pembaruan peran akan gagal karena kesalahan. MalformedPolicyDocument

Tabel berikut mencantumkan penyedia OIDC bersama yang memerlukan kontrol penyedia identitas dalam kebijakan kepercayaan peran dan informasi tambahan untuk membantu Anda mengonfigurasi kontrol penyedia identitas.

IdP OIDC	URL OIDC	Klaim Sewa	Klaim yang Diperlukan
Amazon Cognito	`cognito-identity.amazonaws.com`	aud	`cognito-identity.amazonaws.com:aud`
Sentinel Azure	`https://sts.windows.net/33e01921-4d64-4f8c-a055-5bdaffd5e33d`	`sts:RoleSessionName`	`sts:RoleSessionName`
Buildkite	`https://agent.buildkite.com`	sub	`agent.buildkite.com:sub`
SaaS Codefresh	`https://oidc.codefresh.io`	sub	`oidc.codefresh.io:sub`
Studio DVC	`https://studio.datachain.ai/api`	sub	`studio.datachain.ai/api:sub`

GitHub tindakan	`https://token.actions.githubusercontent.com`	sub atau job_workflow_ref	`token.actions.githubusercontent.com:sub`
			
			atau
			
			`token.actions.githubusercon`

IdP OIDC	URL OIDC	Klaim Sewa	Klaim yang Diperlukan
			`tent.com:job_workflow_ref`
GitHub streaming log audit	`https://oidc-configuration.audit-log.githubusercontent.com`	sub	`oidc-configuration.audit-log.githubusercontent.com:sub`
GitHub vstoken	`https://vstoken.actions.githubusercontent.com`	sub	`vstoken.actions.githubusercontent.com:sub`
GitLab	`https://gitlab.com`	sub	`gitlab.com:sub`

IdP OIDC	URL OIDC	Klaim Sewa	Klaim yang Diperlukan
IBM Turbonomik SaaS*	• https://rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7 • https://rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr • https://rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg • https://rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26	sub	• rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08idd0nt4hekbn7:sub • rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr:sub • rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg:sub • rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26:sub

IdP OIDC	URL OIDC	Klaim Sewa	Klaim yang Diperlukan
	• https://oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb • https://oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu		• oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb:sub • oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu:sub
Awan Pulumi	https://api.pulumi.com/oidc	aud	api.pulumi.com/oidc:aud
sandboxes.cloud	https://sandboxes.cloud	aud	sandboxes.cloud:aud
Scalr	https://scalr.io	sub	scalr.io:sub
Awan Shisho	https://tokens.cloud.shisho.dev	sub	tokens.cloud.shisho.dev:sub
Awan Terraform	https://app.terraform.io	sub	app.terraform.io:sub

IdP OIDC	URL OIDC	Klaim Sewa	Klaim yang Diperluka n
Naik	`https://p roidc.upb ound.io`	sub	`proidc.up bound.io:sub`
Titik akhir global Vercel	`https://o idc.vercel.com`	aud	`oidc.verc el.com:aud`

* IBM Turbonomic secara berkala memperbarui URL Penerbit OIDC mereka dengan versi platform baru. Kami akan menambahkan emiten Turbonomic OIDC tambahan dalam cakupan sebagai penyedia bersama sesuai kebutuhan.

Untuk setiap OIDC baru IdPs yang diidentifikasi IAM sebagai bersama, kontrol penyedia identitas yang diperlukan untuk kebijakan kepercayaan peran akan didokumentasikan dan ditegakkan dengan cara yang sama.

Sumber daya tambahan

Sumber daya tambahan:

- Untuk informasi selengkapnya tentang membuat peran IAM untuk federasi OIDC, lihat. Membuat peran untuk federasi OpenID Connect (konsol)
- Untuk daftar kunci kondisi IAM yang dapat digunakan untuk klaim, lihatKunci yang tersedia untuk AWS Federasi OIDC.

Federasi SAML 2.0

AWS mendukung federasi identitas dengan SAML 2.0 (Security Assertion Markup Language 2.0), standar terbuka yang digunakan banyak penyedia identitas (IdPs). Fitur ini mengaktifkan sistem masuk tunggal federasi (SSO), sehingga pengguna dapat masuk ke Konsol Manajemen AWS atau memanggil operasi AWS API tanpa Anda harus membuat pengguna IAM untuk semua orang di organisasi Anda. Dengan menggunakan SAML, Anda dapat menyederhanakan proses konfigurasi federasi dengan AWS, karena Anda dapat menggunakan layanan IdP alih-alih menulis kode proxy identitas kustom.

> **Note**
>
> Federasi identitas IAM SAML mendukung respons SAML terenkripsi dari penyedia identitas SAML-based federasi (). IdPs Pusat Identitas IAM dan Amazon Cognito tidak mendukung pernyataan SAML terenkripsi dari penyedia identitas IAM SAML.
> Anda dapat secara tidak langsung menambahkan dukungan untuk pernyataan SAML terenkripsi ke federasi kumpulan identitas Amazon Cognito dengan kumpulan pengguna Amazon Cognito. Kumpulan pengguna memiliki federasi SAML yang independen dari federasi IAM SAML dan mendukung penandatanganan dan enkripsi SAML. Meskipun fitur ini tidak meluas langsung ke kumpulan identitas, kumpulan pengguna dapat berupa IdPs kumpulan identitas. Untuk menggunakan enkripsi SAML dengan kumpulan identitas, tambahkan penyedia SAML dengan enkripsi ke kumpulan pengguna yang merupakan IdP ke kumpulan identitas.
> Penyedia SAML Anda harus dapat mengenkripsi pernyataan SAML dengan kunci yang disediakan kumpulan pengguna Anda. Kumpulan pengguna tidak akan menerima pernyataan yang dienkripsi dengan sertifikat yang disediakan IAM.

Federasi IAM mendukung penggunaan kasus ini:

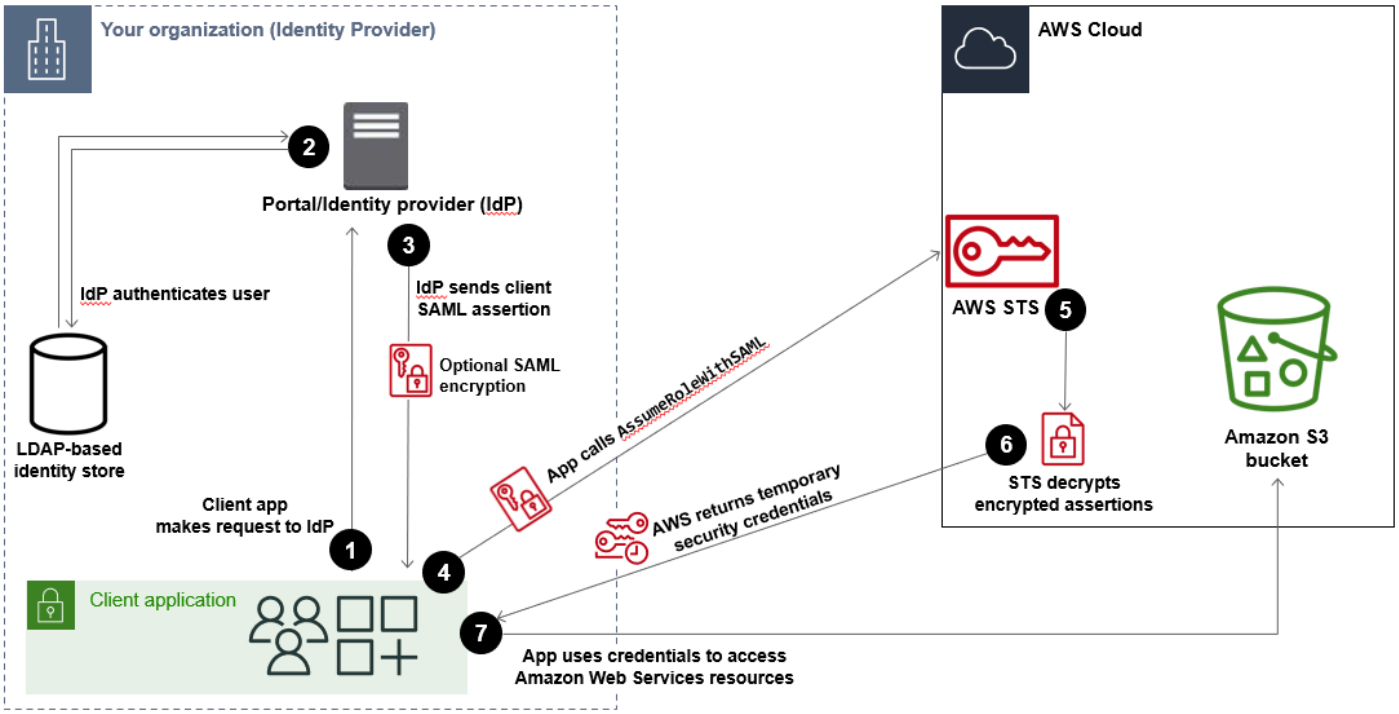
- Akses federasi untuk mengizinkan pengguna atau aplikasi di organisasi Anda memanggil operasi AWS API. Kasus penggunaan ini dibahas di bagian berikut. Anda menggunakan pernyataan SAML (sebagai bagian dari tanggapan autentikasi) yang dibuat dalam organisasi Anda untuk mendapatkan kredensial keamanan sementara. Skenario ini serupa dengan skenario federasi lain yang mendukung IAM, seperti yang dijelaskan dalam Minta kredentif keamanan sementara dan Federasi OIDC. Namun, SAML 2.0 berbasis IdPs di organisasi Anda menangani banyak detail pada waktu berjalan untuk melakukan otentikasi dan pemeriksaan otorisasi.
- Web-basedsingle sign-on (SSO) ke Konsol Manajemen AWS dari organisasi Anda. Pengguna dapat masuk ke portal di organisasi Anda yang dihosting oleh IdP yang kompatibel dengan SAML 2.0, memilih opsi untuk dikunjungi AWS, dan dialihkan ke konsol tanpa harus memberikan informasi masuk tambahan. Anda dapat menggunakan IdP SAML pihak ketiga untuk menetapkan akses SSO ke konsol atau Anda dapat membuat IdP kustom untuk mengaktifkan akses konsol bagi pengguna eksternal Anda. Untuk informasi selengkapnya tentang membangun IdP kustom, lihat Aktifkan akses broker identitas khusus ke AWS konsol.

Topik

- Menggunakan SAML-based federasi untuk akses API ke AWS
- Ikhtisar pengkonfigurasian federasi berbasis SAML 2.0
- Ikhtisar peran untuk memungkinkan SAML-federated akses ke Anda AWS sumber daya
- Mengidentifikasi pengguna secara unik di SAML-based federasi
- Buat penyedia identitas SAML di IAM
- Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim
- Integrasikan penyedia solusi SAML pihak ketiga dengan AWS
- Mengonfigurasikan pernyataan SAML untuk respons autentikasi
- Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS
- Melihat respons SAML di browser Anda

Menggunakan SAML-based federasi untuk akses API ke AWS

Asumsikan bahwa Anda ingin menyediakan cara bagi karyawan untuk menyalin data dari komputer mereka ke folder cadangan. Anda membuat aplikasi yang dapat dijalankan pengguna di komputer mereka. Di bagian belakang, aplikasi membaca dan menulis objek dalam bucket Amazon S3. Pengguna tidak memiliki akses langsung ke AWS. Alih-alih, proses berikut ini digunakan:



1. Pengguna dalam organisasi Anda menggunakan aplikasi klien untuk meminta autentikasi dari IdP organisasi Anda.
2. IdP mengautentikasi pengguna menggunakan penyimpanan identitas organisasi Anda.
3. IdP menyusun pernyataan SAML dengan informasi tentang pengguna dan mengirimkan pertanyaan ke aplikasi klien. Saat Anda mengaktifkan enkripsi SAML untuk IdP IAM SAML Anda, pernyataan ini dienkripsi oleh IdP eksternal Anda.
4. Aplikasi klien memanggil AWS STS `AssumeRoleWithSAML` API, meneruskan ARN penyedia SAML, ARN peran yang akan diasumsikan, dan pernyataan SAML dari IdP. Jika enkripsi diaktifkan, pernyataan yang dilewatkan melalui aplikasi klien tetap dienkripsi saat transit.
5. (Opsional) AWS STS menggunakan kunci pribadi yang Anda unggah dari IdP eksternal untuk mendekripsi pernyataan SAML terenkripsi.
6. Tanggapan API ke aplikasi klien meliputi kredensial keamanan sementara.
7. Aplikasi klien menggunakan kredensial keamanan sementara untuk menghubungi operasi API Amazon S3.

Ikhtisar pengkonfigurasian federasi berbasis SAML 2.0

Sebelum Anda dapat menggunakan federasi berbasis SAML 2.0 seperti yang dijelaskan dalam skenario dan diagram sebelumnya, Anda harus mengonfigurasi IdP organisasi Anda dan Anda Akun AWS untuk saling percaya. Proses umum untuk mengonfigurasikan kepercayaan ini dijelaskan dalam langkah-langkah berikut. Di dalam organisasi, Anda harus memiliki IdP yang mendukung SAML 2.0, seperti Microsoft Active Directory Federation Service (AD FS, bagian dari Windows Server), Shibboleth, atau penyedia SAML 2.0 lain yang kompatibel.

> **Note**
>
> Untuk meningkatkan ketahanan federasi, sebaiknya Anda mengonfigurasi IdP dan AWS federasi untuk mendukung beberapa titik akhir masuk SAML. Untuk detailnya, lihat artikel Blog AWS Keamanan Cara menggunakan titik akhir SAML regional untuk failover.

Konfigurasikan IdP organisasi Anda dan AWS untuk saling percaya

1. Daftar AWS sebagai penyedia layanan (SP) dengan IdP organisasi Anda. Gunakan dokumen metadata SAML dari `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

Untuk daftar *region-code* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir

Anda dapat secara opsional menggunakan dokumen metadata SAML dari`https://signin.aws.amazon.com/static/saml-metadata.xml`.

2. Menggunakan IdP organisasi Anda, Anda menghasilkan file XML metadata SAML yang setara yang dapat menggambarkan IdP Anda sebagai penyedia identitas IAM di. AWS Itu harus menyertakan nama penerbit, tanggal pembuatan, tanggal kedaluwarsa, dan kunci yang AWS dapat digunakan untuk memvalidasi tanggapan otentikasi (pernyataan) dari organisasi Anda.

   Jika Anda mengizinkan pernyataan SAML terenkripsi dikirim dari IdP eksternal Anda, Anda harus membuat file kunci pribadi menggunakan IdP organisasi Anda, dan mengunggah file ini ke konfigurasi IAM SAML Anda dalam format file.pem. AWS STS membutuhkan kunci pribadi ini untuk mendekripsi respons SAML yang sesuai dengan kunci publik yang diunggah ke IdP Anda.

   > **Note**
   >
   > Sebagaimana didefinisikan oleh Profil Interoper V2.0 abilitas Metadata SAML Versi 1.0, IAM tidak mengevaluasi atau mengambil tindakan pada kedaluwarsa X.509 sertifikat dalam dokumen metadata SAML. Jika Anda khawatir tentang X.509 sertifikat yang kedaluwarsa, sebaiknya pantau tanggal kedaluwarsa sertifikat dan merotasi sertifikat sesuai dengan kebijakan tata kelola dan keamanan organisasi Anda.

3. Di konsol IAM, Anda membuat penyedia identitas SAML. Sebagai bagian dari proses ini, Anda mengunggah dokumen metadata SAML dan kunci dekripsi pribadi yang dihasilkan oleh IdP di organisasi Anda di. Step 2 Untuk informasi selengkapnya, lihat Buat penyedia identitas SAML di IAM.
4. Di IAM, Anda membuat satu atau lebih peran IAM. Dalam kebijakan kepercayaan peran, Anda menetapkan penyedia SAML sebagai prinsipal, yang menetapkan hubungan kepercayaan antara organisasi Anda dan AWS. Kebijakan izin peran menetapkan hal yang diizinkan oleh pengguna organisasi Anda di AWS. Untuk informasi selengkapnya, lihat Membuat peran untuk penyedia identitas pihak ketiga.

> **Note**
>
> IDP SAML yang digunakan dalam kebijakan kepercayaan peran harus berada di akun yang sama dengan peran tersebut.

5. Di IdP organisasi Anda, Anda menentukan pernyataan yang memetakan pengguna atau grup dalam organisasi Anda ke peran IAM. Perhatikan bahwa pengguna dan grup yang berbeda pada organisasi Anda mungkin memetakan IAM role yang berbeda. Langkah yang tepat untuk melakukan pemetaan bergantung pada IdP apa yang Anda gunakan. Di skenario sebelumnya dari folder Amazon S3 untuk pengguna, mungkin semua pengguna akan memetakan peran yang sama yang memberikan izin Amazon S3. Untuk informasi selengkapnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

   Jika IdP Anda mengaktifkan SSO ke AWS konsol, maka Anda dapat mengonfigurasi durasi maksimum sesi konsol. Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS.

6. Dalam aplikasi yang Anda buat, Anda memanggil AWS Security Token Service `AssumeRoleWithSAML` API, meneruskannya ARN penyedia SAML yang Anda buatStep 3, ARN peran untuk mengasumsikan bahwa Anda membuatStep 4, dan pernyataan SAML tentang pengguna saat ini yang Anda dapatkan dari IdP Anda. AWS memastikan bahwa permintaan untuk mengambil peran berasal dari IdP yang direferensikan dalam penyedia SAML.

   Untuk informasi selengkapnya, lihat AssumeRoleWithSAML di dalam Referensi API AWS Security Token Service .

7. Jika permintaan berhasil, API mengembalikan serangkaian kredensial keamanan sementara, yang dapat digunakan aplikasi Anda untuk membuat permintaan yang ditandatangani ke AWS. Aplikasi Anda memiliki informasi tentang pengguna saat ini dan dapat mengakses folder khusus pengguna di Amazon S3, seperti yang dijelaskan dalam skenario sebelumnya.

Ikhtisar peran untuk memungkinkan SAML-federated akses ke Anda AWS sumber daya

Peran yang Anda buat di IAM menentukan apa yang diizinkan dilakukan oleh prinsipal federasi SAML dari organisasi Anda. AWS Saat membuat kebijakan kepercayaan untuk peran, Anda menentukan penyedia SAML yang dibuat sebelumnya sebagai `Principal`. Anda juga dapat mencakup kebijakan kepercayaan dengan `Condition` untuk hanya mengizinkan pengguna yang cocok dengan atribut

SAML tertentu untuk mengakses peran. Misalnya, Anda dapat menentukan bahwa hanya pengguna yang afiliasi SAML-nya `staff` (seperti yang ditegaskan olehhttps://openidp.feide.no) yang diizinkan untuk mengakses peran, seperti yang diilustrasikan oleh kebijakan sampel berikut:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://us-east-1.signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws-cn:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:aud": "https://cn-north-1.signin.amazonaws.cn/saml",
                    "saml:iss": "https://openidp.feide.no"
                },
```

```
                "ForAllValues:StringLike": {
                    "saml:edupersonaffiliation": [
                        "staff"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> IDP SAML yang digunakan dalam kebijakan kepercayaan peran harus berada di akun yang sama dengan peran tersebut.

K `saml:aud` unci konteks dalam kebijakan menentukan URL yang ditampilkan browser Anda saat masuk ke konsol. URL titik akhir masuk ini harus sesuai dengan atribut penerima penyedia identitas Anda. Anda dapat menyertakan URL masuk dalam wilayah tertentu. AWS merekomendasikan penggunaan titik akhir Regional alih-alih titik akhir global untuk meningkatkan ketahanan federasi. Jika Anda hanya memiliki satu titik akhir yang dikonfigurasi, Anda tidak akan dapat bergabung ke AWS dalam hal yang tidak mungkin terjadi bahwa titik akhir menjadi tidak tersedia. Untuk daftar *`region-code`* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir

Contoh berikut menunjukkan format URL masuk dengan opsional yang `region-code`.

`https://`*`region-code`*`.signin.aws.amazon.com/saml`

Jika enkripsi SAML diperlukan, URL masuk harus menyertakan pengidentifikasi unik yang ditetapkan AWS untuk penyedia SAML Anda, yang dapat Anda temukan di halaman detail penyedia identitas. Dalam contoh berikut, URL masuk menyertakan pengenal unik IdP, yang mengharuskan /acs/ ditambahkan ke jalur masuk.

`https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

Untuk kebijakan izin dalam peran tersebut, Anda menentukan izin seperti yang Anda inginkan untuk peran apa pun. Misalnya, jika pengguna dari organisasi Anda diizinkan untuk menjalankan instans Amazon Elastic Compute Cloud, Anda harus secara eksplisit mengizinkan tindakan Amazon EC2 dalam kebijakan izin, seperti yang ada di AmazonEC2FullAccess kebijakan yang dikelola.

Untuk informasi selengkapnya tentang kunci SAML yang dapat Anda periksa di kebijakan, lihat Kunci yang tersedia untuk SAML-based AWS STS federasi.

Mengidentifikasi pengguna secara unik di SAML-based federasi

Saat Anda membuat kebijakan akses di IAM, seringkali berguna untuk dapat menentukan izin berdasarkan identitas pengguna. Misalnya, bagi pengguna yang telah difederasikan menggunakan SAML, aplikasi mungkin ingin menyimpan informasi di Amazon S3 menggunakan struktur seperti ini:

```
amzn-s3-demo-bucket/app1/user1
amzn-s3-demo-bucket/app1/user2
amzn-s3-demo-bucket/app1/user3
```

Anda dapat membuat bucket (`amzn-s3-demo-bucket`) dan folder (`app1`) melalui konsol Amazon S3 atau AWS CLI, karena itu adalah nilai statis. Namun, folder khusus pengguna (*`user1user2`*,*`user3`*,, dll.) harus dibuat pada waktu berjalan menggunakan kode, karena nilai yang mengidentifikasi pengguna tidak diketahui sampai pertama kali pengguna masuk melalui proses federasi.

Untuk menulis kebijakan yang merujuk detail spesifik pengguna sebagai bagian dari nama sumber daya, Identitas pengguna harus tersedia dalam kunci SAML yang dapat digunakan dalam ketentuan kebijakan. Kunci-kunci berikut tersedia untuk federasi berbasis SAML 2.0 untuk digunakan dalam kebijakan IAM. Anda dapat menggunakan nilai yang dikembalikan oleh kunci berikut untuk membuat pengidentifikasi pengguna unik untuk sumber daya seperti folder Amazon S3.

- `saml:namequalifier`. Nilai hash berdasarkan konkasi dari `Issuer` nilai tanggapan (`saml:iss`) dan string dengan ID akun `AWS` dan nama ramah (bagian terakhir ARN) penyedia SAML di IAM. Gabungan ID akun dan nama ramah penyedia SAML tersedia untuk kebijakan IAM sebagai kuncinya. `saml:doc` ID akun dan nama penyedia harus dipisahkan dengan '/' seperti pada "123456789012/provider_name”. Untuk informasi lebih lanjut, lihat kunci `saml:doc` di Kunci yang tersedia untuk SAML-based AWS STS federasi.

  Kombinasi `NameQualifier` dan `Subject` dapat digunakan untuk mengidentifikasi prinsipal federasi SAML secara unik. Kode pseudo berikut menunjukkan bagaimana nilai ini dihitung. Dalam pseudocode ini + menunjukkan penggabungan, `SHA1` mewakili fungsi yang menghasilkan ringkasan pesan menggunakan SHA-1, dan `Base64` mewakili fungsi yang menghasilkan versi kode dari Base-64 output hash.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
MySAMLIdP" ) )
```

Untuk informasi selengkapnya tentang kunci kebijakan yang tersedia untuk SAML-based federasi, lihatKunci yang tersedia untuk SAML-based AWS STS federasi.

- `saml:sub` (string). Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna secara individu dalam organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (string). Kunci ini dapat berupa `persistent`, `transient`, atau `Format` URI penuh dari `Subject` dan elemen `NameID` yang digunakan dalam pernyataan SAML Anda. Nilai dari `persistent` menunjukkan bahwa nilai dalam `saml:sub` sama untuk pengguna di semua sesi. Jika nilainya `transient`, pengguna memiliki nilai `saml:sub` untuk setiap sesi. Untuk informasi tentang elemen `NameID` `Format` atribut, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

Contoh berikut menunjukkan kebijakan izin yang menggunakan kunci sebelumnya untuk memberikan izin ke folder khusus pengguna di Amazon S3. Kebijakan ini mengasumsikan bahwa objek Amazon S3 diidentifikasi menggunakan awalan yang menyertakan keduanya `saml:namequalifier` dan`saml:sub`. Perhatikan bahwa elemen `Condition` mencakup pengujian untuk memastikan bahwa `saml:sub_type` diatur ke `persistent`. Jika diatur ke `transient`, nilai `saml:sub` untuk pengguna dapat berbeda untuk setiap sesi, dan kombinasi nilai tidak boleh digunakan untuk mengidentifikasi folder khusus pengguna.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
```

```
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Untuk informasi selengkapnya tentang pernyataan pemetaan dari IdP ke kunci kebijakan, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

Buat penyedia identitas SAML di IAM

Penyedia identitas IAM SAML 2.0 adalah entitas di IAM yang menggambarkan layanan penyedia identitas eksternal (IdP) yang mendukung standar SAML 2.0 (Security Assertion Markup Language 2.0). Anda menggunakan penyedia identitas IAM ketika Anda ingin membangun kepercayaan antara SAML-compatible IdP seperti Shibboleth atau Layanan Federasi Direktori Aktif dan AWS, sehingga pengguna Anda dapat mengakses sumber daya. AWS Penyedia identitas IAM SAML digunakan sebagai prinsipal dalam kebijakan kepercayaan IAM.

Untuk informasi selengkapnya tentang skenario ini, lihat Federasi SAML 2.0.

Anda dapat membuat dan mengelola penyedia identitas IAM di Konsol Manajemen AWS atau dengan AWS CLI, Tools for Windows PowerShell, atau panggilan AWS API.

Setelah Anda membuat penyedia identitas SAML, Anda harus membuat satu atau lebih peran IAM. Peran adalah identitas AWS yang tidak memiliki kredensialnya sendiri (seperti yang dilakukan pengguna). Tetapi dalam konteks ini, peran secara dinamis ditetapkan ke prinsipal federasi SAML yang diautentikasi oleh IdP Anda. Peran ini mengizinkan IdP Anda untuk meminta kredentif keamanan sementara untuk akses ke AWS. Kebijakan yang ditetapkan untuk peran menentukan apa yang diizinkan untuk dilakukan pengguna AWS. Untuk menciptakan peran bagi federasi SAML, lihat Membuat peran untuk penyedia identitas pihak ketiga.

Terakhir, setelah Anda membuat peran, Anda menyelesaikan kepercayaan SAML dengan mengonfigurasi IdP Anda dengan informasi tentang AWS dan peran yang Anda ingin digunakan oleh prinsipal federasi SAML Anda. Ini disebut sebagai mengonfigurasi kepercayaan pihak pengandal antara IdP dan AWS. Untuk mengkonfigurasi kepercayaan pihak yang bergantung, lihat Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim.

Topik

- Prasyarat
- Membuat dan mengelola penyedia identitas IAM SAML (konsol)
- Kelola kunci enkripsi SAML
- Membuat dan mengelola Penyedia Identitas IAM SAML (AWS CLI)
- Membuat dan mengelola penyedia identitas IAM SAML (AWS API)
- Langkah selanjutnya

Prasyarat

Sebelum Anda dapat membuat penyedia identitas SAML, Anda harus memiliki informasi berikut dari IdP Anda.

- Dapatkan dokumen metadata SAML dari IdP Anda. Dokumen ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons autentikasi SAML (penegasan) yang diterima dari IdP. Untuk menghasilkan dokumen metadata, gunakan perangkat lunak manajemen identitas yang disediakan oleh IdP eksternal Anda.

> **Important**
>
> File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons otentikasi SAML (pernyataan) yang diterima dari IdP. File metadata harus dikodekan dalam UTF-8 format tanpa tanda urutan byte (BOM). Untuk menghapus BOM, Anda dapat menyandikan file seperti UTF-8 menggunakan alat pengeditan teks, seperti Notepad ++.
> X.509 Sertifikat yang disertakan sebagai bagian dari dokumen metadata SAML harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, X.509 sertifikat juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali di sertifikat. Jika X.509 sertifikat tidak memenuhi salah satu kondisi, pembuatan IdP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”. Sebagaimana didefinisikan oleh Profil Interoper V2.0 abilitas Metadata SAML Versi 1.0, IAM tidak mengevaluasi atau mengambil tindakan pada kedaluwarsa X.509 sertifikat dalam dokumen metadata SAML. Jika Anda khawatir tentang X.509 sertifikat yang kedaluwarsa, sebaiknya pantau tanggal kedaluwarsa sertifikat dan merotasi sertifikat sesuai dengan kebijakan tata kelola dan keamanan organisasi Anda.

- Ketika Anda memilih untuk mengaktifkan enkripsi SAML, Anda harus membuat file kunci pribadi menggunakan IdP Anda, dan mengunggah file ini ke konfigurasi IAM SAML Anda dalam format file.pem. AWS STS membutuhkan kunci pribadi ini untuk mendekripsi respons SAML yang sesuai dengan kunci publik yang diunggah ke IdP Anda. Algoritma berikut didukung:
  - Algoritme enkripsi
    - AES-128
    - AES-256
    - RSA-OAEP
  - Algoritma transportasi utama
    - AES-CBC
    - AES-GCM

  Lihat dokumentasi penyedia identitas Anda untuk langkah-langkah untuk membuat kunci pribadi.

> **Note**
>
> Pusat Identitas IAM dan Amazon Cognito tidak mendukung pernyataan SAML terenkripsi dari penyedia identitas IAM SAML. Anda dapat secara tidak langsung menambahkan dukungan untuk pernyataan SAML terenkripsi ke federasi kumpulan identitas Amazon Cognito dengan kumpulan pengguna Amazon Cognito. Kumpulan pengguna memiliki federasi SAML yang independen dari federasi IAM SAML dan mendukung penandatanganan dan enkripsi SAML. Meskipun fitur ini tidak meluas langsung ke kumpulan identitas, kumpulan pengguna dapat berupa IdPs kumpulan identitas. Untuk menggunakan enkripsi SAML dengan kumpulan identitas, tambahkan penyedia SAML dengan enkripsi ke kumpulan pengguna yang merupakan IdP ke kumpulan identitas. Penyedia SAML Anda harus dapat mengenkripsi pernyataan SAML dengan kunci yang disediakan kumpulan pengguna Anda. Kumpulan pengguna tidak akan menerima pernyataan yang dienkripsi dengan sertifikat yang disediakan IAM.

Untuk petunjuk tentang cara mengonfigurasi banyak yang tersedia IdPs untuk digunakan AWS, termasuk cara menghasilkan dokumen metadata SAML yang diperlukan, lihatIntegrasikan penyedia solusi SAML pihak ketiga dengan AWS.

Untuk bantuan dengan federasi SAML, lihat Memecahkan masalah federasi SAML.

Membuat dan mengelola penyedia identitas IAM SAML (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk membuat, memperbarui, dan menghapus penyedia identitas IAM SAML. Untuk bantuan dengan federasi SAML, lihat Memecahkan masalah federasi SAML.

Untuk membuat penyedia identitas IAM SAML (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Penyedia identitas lalu pilih Tambahkan penyedia.
3. Untuk Konfigurasi penyedia, pilih SAML.
4. Ketikkan nama untuk penyedia identitas.
5. Untuk Dokumen metadata, pilih Pilih file, tentukan dokumen metadata SAML yang Anda unduh di the section called “Prasyarat”.

> **Note**
>
> Tanggal Valid hingga ditampilkan untuk penyedia identitas ditetapkan oleh IAM pada waktu pembuatan penyedia. Itu tidak mencerminkan `cacheDuration` atribut `validUntil` or dalam dokumen metadata SAML Anda. IAM tidak menggunakan atribut metadata ini untuk menentukan Valid hingga tanggal.
> IAM tidak mengevaluasi atau mengambil tindakan pada kedaluwarsa dokumen metadata SAML atau X.509 sertifikat dalam dokumen metadata SAML selama otentikasi SAML. Validitas otentikasi SAML diberlakukan pada tingkat pernyataan, menggunakan `NotOnOrAfter` atribut dalam pernyataan `Conditions` dan elemen. `SubjectConfirmationData` Jika Anda khawatir tentang X.509 sertifikat yang kedaluwarsa, sebaiknya pantau tanggal kedaluwarsa sertifikat dan merotasi sertifikat sesuai dengan kebijakan tata kelola dan keamanan organisasi Anda.

6. (Opsional) Untuk enkripsi SAML, pilih Pilih file dan pilih file kunci pribadi yang Anda buat. the section called “Prasyarat” Pilih M emerlukan enkripsi untuk menerima hanya permintaan terenkripsi dari IdP Anda.
7. (Opsional) Untuk Tambah tag, Anda dapat menambahkan pasangan kunci-nilai untuk membantu mengidentifikasi dan mengatur. IdPs Anda juga dapat menggunakan tanda untuk mengontrol akses ke sumber daya AWS . Untuk mempelajari selengkapnya tentang penandaan penyedia identitas, lihat Kata kunci IAM SAML identity provider.

   Pilih Tambahkan tanda. Masukkan nilai untuk setiap pasangan nilai-kunci tanda.
8. Verifikasi informasi yang telah Anda berikan. Setelah selesai, pilih Tambahkan penyedia.
9. Tetapkan peran IAM ke penyedia identitas Anda. Peran ini memberikan identitas pengguna eksternal yang dikelola oleh penyedia identitas Anda izin untuk mengakses AWS sumber daya di akun Anda. Untuk mempelajari lebih lanjut tentang menciptakan peran untuk federasi identitas, lihat Membuat peran untuk penyedia identitas pihak ketiga.

> **Note**
>
> IDP SAML yang digunakan dalam kebijakan kepercayaan peran harus berada di akun yang sama dengan peran tersebut.

Untuk menghapus penyedia SAML (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Penyedia identitas.
3. Pilih tombol radio di samping penyedia identitas yang ingin Anda hapus.
4. Pilih Hapus. Jendela baru terbuka.
5. Konfirmasikan bahwa Anda ingin menghapus penyedia dengan mengetik kata `delete` di bidangnya. Lalu, pilih Hapus.

Kelola kunci enkripsi SAML

Anda dapat mengonfigurasi penyedia IAM SAML untuk menerima pernyataan terenkripsi dalam respons SAML dari IdP eksternal Anda. Pengguna dapat mengambil peran dengan pernyataan SAML terenkripsi AWS dengan menelepon. `sts:AssumeRoleWithSAML`

Enkripsi SAML memastikan bahwa pernyataan aman ketika dilewatkan melalui perantara atau pihak ketiga. Selain itu, fitur ini membantu Anda memenuhi FedRAMP atau persyaratan kebijakan kepatuhan internal apa pun yang mewajibkan pernyataan SAML untuk dienkripsi.

Untuk mengonfigurasi penyedia identitas IAM SAML, lihat. Buat penyedia identitas SAML di IAM Untuk bantuan dengan federasi SAML, lihat Memecahkan masalah feder asi SAML.

Putar kunci enkripsi SAML

IAM menggunakan kunci pribadi yang Anda unggah ke penyedia IAM SAML untuk mendekripsi pernyataan SAML terenkripsi dari IdP Anda. Anda dapat menyimpan hingga dua file kunci pribadi untuk setiap penyedia identitas, memungkinkan Anda memutar kunci pribadi seperlunya. Ketika dua file disimpan, setiap permintaan pertama-tama akan mencoba mendekripsi dengan tanggal Dit ambahkan terbaru, kemudian IAM mencoba mendekripsi permintaan dengan tanggal Dit ambahkan tertua.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Penyedia identitas dan kemudian pilih penyedia Anda dari daftar.
3. Pilih tab enkripsi SAML dan pilih Tambahkan kunci baru.
4. Pilih Pilih file dan unggah kunci pribadi yang Anda unduh dari IdP Anda sebagai file.pem. Kemudian, pilih Tambah kunci.

5. Di bagian K unci pribadi untuk dekripsi SAML, pilih file kunci pribadi yang kedaluwarsa dan pilih Hapus. Kami menyarankan Anda menghapus kunci pribadi yang kedaluwarsa setelah menambahkan kunci pribadi baru untuk memastikan upaya pertama untuk mendekripsi pernyataan Anda berhasil.

Membuat dan mengelola Penyedia Identitas IAM SAML (AWS CLI)

Anda dapat menggunakan AWS CLI untuk membuat, memperbarui, dan menghapus penyedia SAML. Untuk bantuan dengan federasi SAML, lihat Memecahkan masalah feder asi SAML.

Untuk membuat penyedia identitas IAM dan mengunggah dokumen metadata (AWS CLI)

- Jalankan perintah ini: `aws iam create-saml-provider`

Untuk memperbarui penyedia identitas IAM SAML (AWS CLI)

Anda dapat memperbarui file metadata, pengaturan enkripsi SAML, dan memutar file dekripsi kunci pribadi untuk penyedia IAM SAML Anda. Untuk memutar kunci pribadi, tambahkan kunci pribadi baru Anda dan kemudian hapus kunci lama dalam permintaan terpisah. Untuk informasi selengkapnya tentang memutar kunci pribadi, lihatKelola kunci enkripsi SAML.

- Jalankan perintah ini:`aws iam update-saml-provider`

Untuk menandai penyedia identitas IAM yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam tag-saml-provider`

Untuk mencantumkan tag untuk penyedia identitas IAM yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam list-saml-provider-tags`

Untuk menghapus tag pada penyedia identitas IAM yang ada (AWS CLI)

- Jalankan perintah ini:`aws iam untag-saml-provider`

Untuk menghapus penyedia identitas IAM SAML (AWS CLI)

1. (Opsional) Untuk mencantumkan informasi untuk semua penyedia, seperti ARN, tanggal pembuatan, dan kedaluwarsa, jalankan perintah berikut:

   - `aws iam list-saml-providers`

2. (Opsional) Untuk mendapatkan informasi tentang penyedia tertentu, seperti ARN, tanggal pembuatan, tanggal kedaluwarsa, pengaturan enkripsi, dan informasi kunci pribadi, jalankan perintah berikut:

   - `aws iam get-saml-provider`

3. Untuk menghapus penyedia identitas IAM, jalankan perintah berikut:

   - `aws iam delete-saml-provider`

Membuat dan mengelola penyedia identitas IAM SAML (AWS API)

Anda dapat menggunakan AWS API untuk membuat, memperbarui, dan menghapus penyedia SAML. Untuk bantuan dengan federasi SAML, lihat Memecahkan masalah feder asi SAML.

Untuk membuat penyedia identitas IAM dan mengunggah dokumen metadata (AWS API)

- Hubungi operasi ini: `CreateSAMLProvider`

Untuk memperbarui penyedia identitas IAM SAML (AWS API)

Anda dapat memperbarui file metadata, pengaturan enkripsi SAML, dan memutar file dekripsi kunci pribadi untuk penyedia IAM SAML Anda. Untuk memutar kunci pribadi, tambahkan kunci pribadi baru Anda dan kemudian hapus kunci lama dalam permintaan terpisah. Untuk informasi selengkapnya tentang memutar kunci pribadi, lihatKelola kunci enkripsi SAML.

- Hubungi operasi ini: `UpdateSAMLProvider`

Untuk menandai penyedia identitas IAM yang ada (AWS API)

- Hubungi operasi ini: `TagSAMLProvider`

Untuk mencantumkan tag untuk penyedia identitas IAM yang ada (AWS API)

- Hubungi operasi ini: `ListSAMLProviderTags`

Untuk menghapus tag pada penyedia identitas IAM yang ada (AWS API)

- Hubungi operasi ini: `UntagSAMLProvider`

Untuk menghapus penyedia identitas IAM (AWS API)

1. (Opsional) Untuk mencantumkan informasi untuk semua IdPs, seperti ARN, tanggal pembuatan, dan kedaluwarsa, panggil operasi berikut:
   - `ListSAMLProviders`
2. (Opsional) Untuk mendapatkan informasi tentang penyedia tertentu, seperti ARN, tanggal pembuatan, tanggal kedaluwarsa, pengaturan enkripsi, dan informasi kunci pribadi, panggil operasi berikut:
   - `GetSAMLProvider`
3. Untuk menghapus IdP, panggil operasi berikut:
   - `DeleteSAMLProvider`

Langkah selanjutnya

Setelah Anda membuat penyedia identitas SAML, atur kepercayaan pihak yang bergantung dengan IdP Anda. Anda juga dapat menggunakan klaim dari respons otentikasi IdP Anda dalam kebijakan untuk mengontrol akses ke peran.

- Anda harus memberi tahu IdP tentang AWS sebagai penyedia layanan. Ini disebut menambahkan kepercayaan pihak yang mengandalkan antara IdP Anda dan AWS. Proses yang tepat untuk menambahkan kepercayaan pihak yang bergantung tergantung pada IdP yang Anda gunakan. Lihat perinciannya di Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim.
- Ketika IdP mengirimkan respons yang berisi klaim ke AWS, banyak klaim yang masuk dipetakan ke kunci AWS konteks. Anda dapat menggunakan kunci konteks ini dalam kebijakan IAM menggunakan elemen Kondisi untuk mengontrol akses ke peran. Untuk detailnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi

Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim

Saat Anda membuat penyedia identitas IAM dan peran untuk akses SAML, Anda memberi tahu AWS tentang penyedia identitas eksternal (IdP) dan apa yang diizinkan untuk dilakukan penggunanya. Langkah Anda selanjutnya adalah memberi tahu IdP tentang AWS sebagai penyedia layanan. Ini disebut menambahkan kepercayaan pihak yang bergantung antara IdP dan AWS. Proses yang tepat untuk menambahkan kepercayaan pihak yang bergantung tergantung pada IdP yang Anda gunakan. Untuk detailnya, lihat dokumentasi untuk perangkat lunak manajemen identitas Anda.

Banyak yang IdPs memungkinkan Anda untuk menentukan URL dari mana IdP dapat membaca dokumen XML yang berisi informasi dan sertifikat pihak yang bergantung. Untuk AWS, gunakan URL titik akhir masuk. Contoh berikut menunjukkan format URL dengan opsional yang `region-code`.

`https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

Jika enkripsi SAML diperlukan, URL harus menyertakan pengenal unik yang AWS ditetapkan untuk penyedia SAML Anda, yang dapat Anda temukan di halaman detail penyedia identitas. Contoh berikut menunjukkan URL masuk regional yang menyertakan pengidentifikasi unik.

`https://region-code.signin.aws.amazon.com/static/saml/IdP-ID/saml-metadata.xml`

Untuk daftar *`region-code`* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir Untuk nil AWS ainya, Anda juga dapat menggunakan titik akhir non-regional. `https://signin.aws.amazon.com/saml`

Jika Anda tidak dapat menentukan URL secara langsung, unduh dokumen XML dari URL sebelumnya dan impor ke perangkat lunak IdP Anda.

Anda juga perlu membuat aturan klaim yang sesuai di IdP Anda yang ditentukan AWS sebagai pihak yang mengandalkan. Ketika IdP mengirimkan respons SAML ke AWS titik akhir, itu menyertakan pernyataan SAML yang berisi satu atau lebih klaim. Klaim adalah informasi tentang pengguna dan grupnya. Aturan klaim memetakan informasi tersebut ke dalam atribut SAML. Ini memungkinkan Anda memastikan bahwa respons otentikasi SAML dari IdP berisi atribut yang diperlukan yang AWS digunakan dalam kebijakan IAM untuk memeriksa izin untuk prinsipal federasi SAML. Untuk informasi selengkapnya, lihat topik berikut:

- Ikhtisar peran untuk memungkinkan SAML-federated akses ke Anda AWS sumber daya. Topik ini membahas penggunaan SAML-specific kunci dalam kebijakan IAM dan cara menggunakannya untuk membatasi izin untuk prinsipal federasi SAML.
- Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Topik ini membahas cara mengonfigurasi klaim SAML yang mencakup informasi tentang pengguna. Klaim dikemas menjadi pernyataan SAML dan disertakan dalam respons SAML yang dikirimkan ke AWS. Anda harus memastikan bahwa informasi yang dibutuhkan oleh AWS kebijakan disertakan dalam pernyataan SAML dalam bentuk yang AWS dapat dikenali dan digunakan.
- Integrasikan penyedia solusi SAML pihak ketiga dengan AWS. Topik ini menyediakan tautan ke dokumentasi yang disediakan oleh organisasi pihak ketiga tentang cara mengintegrasikan solusi identitas dengan AWS.

> **Note**
>
> Untuk meningkatkan ketahanan federasi, sebaiknya Anda mengonfigurasi IdP dan AWS federasi untuk mendukung beberapa titik akhir masuk SAML. Untuk detailnya, lihat artikel Blog AWS Keamanan Cara menggunakan titik akhir SAML regional untuk failover.

Integrasikan penyedia solusi SAML pihak ketiga dengan AWS

> **Note**
>
> Kami menyarankan Anda meminta pengguna manusia untuk menggunakan kredenSIAL sementara saat mengakses AWS. Sudahkah Anda mempertimbangkan untuk menggunakan AWS IAM Identity Center? Anda dapat menggunakan Pusat Identitas IAM untuk secara terpusat mengelola akses ke beberapa Akun AWS dan menyediakan akses masuk MFA-protected tunggal kepada pengguna ke semua akun yang ditugaskan dari satu tempat. Dengan IAM Identity Center, Anda dapat membuat dan mengelola identitas pengguna di Pusat Identitas IAM atau dengan mudah terhubung ke penyedia identitas yang kompatibel dengan SAML 2.0 yang ada. Untuk informasi selengkapnya, lihat Apa itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Tautan berikut membantu Anda mengonfigurasi solusi penyedia identitas (IdP) SAML 2.0 pihak ketiga untuk bekerja dengan AWS federasi. Periksa dengan penyedia identitas Anda untuk menentukan

apakah mereka mendukung enkripsi token SAML. Untuk persyaratan enkripsi SAML, lihatKelola kunci enkripsi SAML.

> **Tip**
>
> AWS Teknisi dukungan dapat membantu pelanggan yang memiliki rencana dukungan bisnis dan perusahaan dengan beberapa tugas integrasi yang melibatkan perangkat lunak pihak ketiga. Untuk daftar terkini dari platform dan aplikasi yang didukung, lihat Apa perangkat lunak pihak ketiga yang didukung? dalam FAQ Dukungan AWS .

Solusi	Informasi lebih lanjut
Auth0	Integrasikan dengan Amazon Web Services — Halaman ini di situs web dokumentasi Auth0 memiliki tautan ke sumber daya yang menjelaskan cara mengatur single sign-on (SSO) dengan Konsol Manajemen AWS dan menyertakan contoh. JavaScript Anda dapat mengonfig urasikan Auth0 untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Auth0 Mengumumkan Kemitraan dengan AWS untuk Tag Sesi IAM.
Microsoft Entry	Tutorial: Integrasi Microsoft Entra SSO dengan AWS Single-Account Access - Tutorial di situs web Microsoft ini menjelaskan cara mengatur Microsoft Entra (sebelumnya dikenal sebagai Azure AD) sebagai penyedia identitas (IdP) menggunakan federasi SAML.
Centrify	Konfigurasikan Centrify dan Gunakan SAML untuk SSO ke AWS — Halaman ini di situs web Centrify menjelaskan cara mengonfigurasi Centrify untuk menggunakan SAML untuk SSO. AWS
CyberArk	Konfigur CyberArk asikan untuk menyediakan akses Amazon Web Services (AWS) kepada pengguna yang masuk melalui SAML single sign-on (SSO) dari Portal Pengguna. CyberArk

Solusi	Informasi lebih lanjut
ForgeRock	Platform ForgeRock Identitas terintegrasi dengan AWS. Anda dapat mengonfigurasikan ForgeRock untuk meneruskan tag sesi. Untuk informasi lebih lanjut, lihat Kontrol Akses Berbasis Atribut untuk Amazon Web Services.
Google Workspace	Aplikasi cloud Amazon Web Services — Artikel di situs Bantuan Admin Google Workspace ini menjelaskan cara mengonfigurasi Google Workspace sebagai IdP SAML 2.0 AWS sebagai penyedia layanan.
IBM	Anda dapat mengonfigurasikan IBM untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat IBM Cloud Identity IDAAs salah satu yang pertama mendukung tag AWS sesi.
JumpCloud	Memberikan Akses melalui Peran IAM untuk Masuk Tunggal (SSO) dengan Amazon AWS — Artikel di JumpCloud situs web ini menjelaskan cara mengatur dan mengaktifkan SSO berdasarkan peran IAM untuk. AWS
Matrix42	MyWorkspace Panduan Memulai - Panduan ini menjelaskan cara mengintegrasikan layanan AWS identitas dengan Matrix42 MyWorkspace.

Solusi	Informasi lebih lanjut
Microsoft Active Directory Federation Services (AD FS)	Catatan Bidang: Mengintegrasikan Layanan Federasi Direktori Aktif dengan AWS IAM Identity Center - Posting ini di Blog AWS Arsitektur menjelaskan aliran otentikasi antara AD FS dan AWS IAM Identity Center (Pusat Identitas IAM). IAM Identity Center mendukung federasi identitas dengan SAML 2.0, memungkinkan integrasi dengan solusi AD FS. Pengguna dapat masuk ke portal Pusat Identitas IAM dengan kredenSIAL perusahaan mereka, mengurangi overhead admin untuk mempertahankan kredenSIAL terpisah di Pusat Identitas IAM. Anda juga dapat mengonfigurasi AD FS untuk meneruskan tag sesi. Untuk informasi lebih lanjut, lihat Gunakan kontrol akses berbasis atribut dengan AD FS untuk menyederhanakan manajemen izin IAM.
miniOrange	SSO untuk AWS - Halaman ini di situs web MiniOrange menjelaskan cara membuat akses aman AWS untuk perusahaan dan kontrol penuh atas akses aplikasi. AWS
Okta	Mengintegrasikan Antarmuka Baris Perintah Amazon Web Services Menggunakan Okta - Dari halaman ini di situs dukungan Okta, Anda dapat mempelajari cara mengonfigurasi Okta untuk digunakan dengan. AWS Anda dapat mengonfigurasikan Okta untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Okta dan AWS Mitra untuk Menyederhanakan Akses Melalui Tag Sesi.
Okta	AWS Federasi Akun — Bagian ini di situs web Okta menjelaskan cara mengatur dan mengaktifkan Pusat Identitas IAM untuk. AWS

Solusi	Informasi lebih lanjut
OneLogin	Dari Kno OneLogin wledgebase, **SAML AWS** cari daftar artikel yang menjelaskan cara mengatur fungsionalitas Pusat Identitas IAM antara OneLogin dan AWS untuk skenario peran tunggal dan multi-peran. Anda dapat mengonfigurasi OneLogin untuk mener uskan tag sesi. Untuk informasi selengkapnya, lihat OneLogin dan Tag Sesi: Kontrol Attribute-Based Akses untuk AWS Sumber Daya.
Identitas Ping	PingFederate AWS Konektor — Lihat detail tentang PingFederate AWS Konektor, template koneksi cepat untuk mengatur koneksi masuk tunggal (SSO) dan penyediaan dengan mudah. Baca dokumentasi dan unduh PingFeder ate AWS Konektor terbaru untuk integrasi dengan AWS. Anda dapat mengonfigurasikan Identitas Ping untuk meneruskan tag sesi. Untuk informasi selengkapnya, lihat Meng umumkan Dukungan Identitas Ping untuk Kontrol Attribute-Based Akses di AWS.
RadiantLogic	Mitra Teknologi Radiant Logic - Layanan Identitas RadiantOne Federasi Radiant Logic terintegrasi dengan AWS untuk menyediakan hub identitas untuk SAML-based SSO.
RSA	Amazon Web Services - Panduan Implementasi Siap RSA memberikan panduan untuk mengintegrasikan AWS dan RSA. Untuk informasi selengkapnya tentang konfigurasi SAML, lihat Amazon Web Services - Konfigurasi SSO Halaman Saya SAML - Panduan Implementasi Siap RSA.
Salesforce.com	Cara mengonfigurasi SSO dari Salesforce ke AWS - Artikel panduan ini di situs Salesforce.com pengemban g menjelaskan cara mengatur penyedia identitas (IdP) di Salesforce dan mengkonfigurasi AWS sebagai penyedia layanan.

Solusi	Informasi lebih lanjut
SecureAuth	AWS - SecureAuth SAML SSO - Artikel di SecureAuth situs web ini menjelaskan cara mengatur integrasi SAML AWS untuk alat. SecureAuth
Shibboleth	Cara Menggunakan Shibboleth untuk SSO ke Konsol Manajemen AWS - Entri ini di Blog AWS Keamanan menyediakan tutorial langkah demi langkah tentang cara mengatur Shibboleth dan mengkonfigurasinya sebagai penyedia identitas untuk. AWS Anda dapat mengonfig urasikan Shibboleth untuk meneruskan tag sesi.

Untuk detail selengkapnya, lihat halaman Mit ra IAM di AWS situs web.

Mengonfigurasikan pernyataan SAML untuk respons autentikasi

Setelah Anda memverifikasi identitas pengguna di organisasi Anda, penyedia identitas eksternal (IdP) mengirimkan respons otentikasi ke URL titik AWS akhir masuk. Respons ini adalah permintaan POST yang menyertakan token SAML yang mematuhi standar Pengikatan HTTP POST untuk SAML 2.0 dan berisi elemen, atau klaim berikut. Anda mengonfigurasi klaim ini di SAML-compatible IdP Anda. Lihat dokumentasi IdP Anda untuk instruksi tentang cara memasukkan klaim ini.

Ketika IdP mengirimkan respons yang berisi klaim ke AWS, banyak klaim yang masuk dipetakan ke kunci AWS konteks. Kunci konteks ini dapat diperiksa di kebijakan IAM menggunakan elemen `Condition`. Daftar pemetaan yang tersedia mengikuti di bagian ini Memetakan atribut SAML ke AWS kunci konteks kebijakan kepercayaan.

`Subjek` dan **`NamaID`**

Respons harus menyertakan tepat satu `SubjectConfirmation` `SubjectConfirmationData` elemen dengan elemen yang mencakup `NotOnOrAfter` atribut dan `Recipient` atribut. Atribut Penerima harus menyertakan nilai yang cocok dengan AWS URL titik akhir masuk. IdP Anda dapat menggunakan istilahACS,`Recipient`, atau `Target` untuk merujuk pada atribut ini.

Jika enkripsi SAML diperlukan, URL masuk harus menyertakan pengidentifikasi unik yang ditetapkan AWS untuk penyedia SAML Anda, yang dapat Anda temukan di halaman detail penyedia identitas. Contoh berikut menunjukkan format URL masuk dengan opsional yang `region-code`.

`https://region-code.signin.aws.amazon.com/saml`

Dalam contoh berikut, URL masuk menyertakan pengenal unik, yang mengharuskan /acs/ ditambahkan ke jalur masuk.

`https://region-code.signin.aws.amazon.com/saml/acs/IdP-ID`

Untuk daftar *region-code* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir Untuk nil AWS ainya, Anda juga dapat menggunakan titik akhir `https://signin.aws.amazon.com/saml` masuk global.

`NameID`Delemen dapat memiliki nilai persisten, sementara, atau terdiri dari URI Format lengkap seperti yang disediakan oleh solusi IdP. Nilai persisten menunjukkan bahwa nilai dalam `NameID` sama untuk pengguna di antara sesi. Jika nilainya bersifat sementara, pengguna memiliki `NameID` nilai yang berbeda untuk setiap sesi. Interaksi masuk tunggal mendukung jenis pengidentifikasi berikut:

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

Kebijakan berikut menunjukkan sebuah contoh. Gantikan nilai Anda sendiri untuk nilai yang ditandai.

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://region-code.signin.aws.amazon.com/saml/SAMLSP4SHN3UIS2D558H46"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> K `saml:aud` unci konteks berasal dari atribut penerima SAML karena itu adalah SAML yang setara dengan bidang audiens OIDC, misalnya,. `accounts.google.com:aud`

`PrincipalTag` Atribut SAML

(Opsional) Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Elemen ini memungkinkan Anda untuk meneruskan atribut sebagai tanda sesi dalam pernyataan SAML. Untuk informasi lebih lanjut tentang tag sesi, lihat Lewati tag sesi di AWS STS.

Untuk menyampaikan atribut sebagai tag sesi, sertakan elemen `AttributeValue` yang menentukan nilai tag. Misalnya, untuk melewati pasangan nilai kunci tag `Project` = `Marketing` dan `CostCenter` = `12345`, gunakan atribut berikut. Termasuk elemen `Attribute` terpisah untuk setiap tag.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Untuk mengatur tanda di atas sebagai transitif, sertakan elemen `Attribute` lain dengan atribut `Name` yang diatur ke `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Ini adalah atribut multinilai opsional yang menetapkan tanda sesi Anda sebagai transitif. Tanda transitif tetap ada saat Anda menggunakan sesi SAML untuk mengambil peran lain di AWS. Ini dikenal sebagai perangkaian peran. Misalnya, untuk mengatur tag `Principal` dan `CostCenter` yang bersifat transitif, gunakan atribut berikut untuk menentukan kunci.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

`Atri` but peran SAML

Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/Role`. Elemen ini berisi satu atau lebih `AttributeValue` elemen yang mencantumkan penyedia identitas IAM dan peran tempat pengguna dipetakan oleh IdP Anda. Peran IAM dan penyedia identitas IAM ditentukan sebagai pasangan ARN yang dipisahkan koma dalam format yang sama dengan parameter and yang diteruskan ke`RoleArn`. `PrincipalArn` AssumeRoleWithSAML Elemen ini harus memuat setidaknya satu pasangan peran-penyedia (elemen `AttributeValue`), dan dapat berisi beberapa pasangan. Jika elemen berisi beberapa pasangan, maka pengguna diminta untuk memilih peran mana yang akan diambil ketika mereka menggunakan WebSSO untuk masuk ke. Konsol Manajemen AWS

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/Role` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

`RoleSessionName` Atribut SAML

Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. Elemen ini berisi satu elemen `AttributeValue` yang menyediakan pengenal untuk kredensial sementara yang dikeluarkan ketika peran diambil. Anda dapat menggunakan ini untuk mengasosiasikan kredensial sementara dengan pengguna yang menggunakan aplikasi Anda. Elemen ini digunakan untuk menampilkan informasi pengguna di Konsol Manajemen AWS. Nilai dalam elemen `AttributeValue` harus terdiri dari 2 hingga 64 karakter, dapat berisi hanya karakter alfanumerik, garis bawah, dan karakter berikut: . , + = @ - (tanda hubung). Tidak dapat berisi spasi. Nilai ini umumnya adalah ID pengguna (`john`) atau alamat email (`johndoe@example.com`). Nilai tidak boleh berupa nilai yang mencakup spasi, seperti nama tampilan pengguna (`John Doe`).

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/RoleSessionName` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

`SessionDuration` Atribut SAML

(Opsional) Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/SessionDuration`. Elemen ini berisi satu `AttributeValue` elemen yang menentukan berapa lama pengguna dapat mengakses Konsol Manajemen AWS sebelum harus meminta kredenSIAL sementara baru. Nilai adalah bilangan bulat yang mewakili jumlah detik untuk sesi. Nilai dapat berkisar dari 900 detik (15 menit) hingga 43200 detik (12 jam). Jika atribut ini tidak ada, maka kredensial berlangsung selama satu jam (nilai default parameter `DurationSeconds` dari API `AssumeRoleWithSAML`).

Untuk menggunakan atribut ini, Anda harus mengonfigurasi penyedia SAML untuk menyediakan akses masuk tunggal ke titik akhir web masuk Konsol Manajemen AWS melalui konsol di. `https://region-code.signin.aws.amazon.com/saml` Untuk daftar *`region-code`* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir Anda dapat menggunakan URL berikut secara opsional:`https://signin.aws.amazon.com/static/saml`. Perhatikan bahwa atribut ini hanya memperluas sesi ke Konsol Manajemen AWS. Atribut ini tidak dapat memperpanjang masa pakai kredensial lainnya. Namun, jika dibuat dalam Panggilan API `AssumeRoleWithSAML`, atribut dapat digunakan untuk mempersingkat durasi sesi. Masa pakai kredensial default yang dikembalikan oleh panggilan adalah 60 menit.

Perhatikan juga bahwa jika atribut `SessionNotOnOrAfter` juga ditentukan, nilai yang lebih sedikit dari kedua atribut, `SessionDuration` atau `SessionNotOnOrAfter`, menetapkan durasi maksimum sesi konsol.

Saat Anda mengaktifkan sesi konsol dengan durasi yang diperpanjang, risiko penyusupan kredensial meningkat. Untuk membantu Anda mengurangi risiko ini, Anda dapat segera menonaktifkan sesi konsol aktif untuk peran apa pun dengan memilih Mencabut Sesi pada halaman Ringkasan Peran

di konsol IAM. Untuk informasi selengkapnya, lihat Mencabut kredensi keamanan sementara peran IAM.

> **Important**
>
> Nilai dari atribut `Name` dalam tag `Attribute` peka terhadap huruf besar-kecil. Nilai harus diatur ke `https://aws.amazon.com/SAML/Attributes/SessionDuration` dengan tepat.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

`SourceIdentity` Atribut SAML

(Opsional) Anda dapat menggunakan elemen `Attribute` dengan atribut `Name` diatur ke `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Elemen ini berisi satu `AttributeValue` elemen yang menyediakan pengenal untuk orang atau aplikasi yang menggunakan peran IAM. Nilai untuk identitas sumber tetap ada saat Anda menggunakan sesi SAML untuk mengambil peran lain yang AWS dikenal sebagai rantai ??? peran. Nilai untuk identitas sumber hadir dalam permintaan untuk setiap tindakan yang diambil selama sesi peran. Nilai yang ditetapkan tidak dapat diubah selama sesi peran. Administrator kemudian dapat menggunakan AWS CloudTrail log untuk memantau dan mengaudit informasi identitas sumber untuk menentukan siapa yang melakukan tindakan dengan peran bersama.

Nilai dalam `AttributeValue` elemen harus antara 2 dan 256 karakter, hanya dapat berisi karakter alfanumerik, garis bawah, dan karakter berikut:., + = @ - (tanda hubung). Tidak dapat berisi spasi. Nilai biasanya merupakan atribut yang berhubungan dengan pengguna seperti id pengguna (`john`) atau alamat email (`johndoe@example.com`). Nilai tidak boleh berupa nilai yang mencakup spasi, seperti nama tampilan pengguna (`John Doe`). Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

> **Important**
>
> Jika pernyataan SAML Anda dikonfigurasikan untuk menggunakan atribut `SourceIdentity`, kebijakan kepercayaan Anda juga harus mencakup tindakan

`sts:SetSourceIdentity`, atau operasi pengambilan peran akan gagal. Untuk informasi selengkapnya tentang penggunaan identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Untuk meneruskan atribut identitas sumber, sertakan elemen `AttributeValue` yang menentukan nilai identitas sumber. Misalnya, untuk meneruskan identitas sumber `Diego` menggunakan atribut berikut.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>Diego</AttributeValue>
```

Memetakan atribut SAML ke AWS kunci konteks kebijakan kepercayaan

Tabel di bagian ini mencantumkan atribut SAML yang umum digunakan dan cara mereka dipetakan untuk memercayai kunci konteks syarat kebijakan di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukannya, bandingkan kunci dengan nilai yang disertakan dalam pernyataan yang menyertai permintaan akses SAML.

Important

Kunci ini hanya tersedia di kebijakan kepercayaan IAM (kebijakan yang menentukan siapa yang dapat mengambil peran) dan tidak berlaku untuk kebijakan izin.

Di tabel atribut eduPerson dan eduOrg, nilai diketik sebagai string atau sebagai daftar string. Untuk nilai string, Anda dapat menguji nilai ini dalam kebijakan kepercayaan IAM menggunakan ketentuan `StringEquals` atau `StringLike`. Untuk nilai yang berisi daftar string, Anda dapat menggunakan `ForAnyValue` dan `ForAllValues` operator set kebijakan untuk menguji nilai-nilai dalam kebijakan kepercayaan.

Note

Anda harus menyertakan hanya satu klaim per kunci AWS konteks. Jika Anda menyertakan lebih dari satu, hanya satu klaim yang akan dipetakan.

Tabel berikut menunjukkan atribut EduPerson dan EduOrg.

Atribut eduPerson atau eduOrg (kunci **`Name`**)	Memetakan ke kunci AWS konteks ini (**`FriendlyName`** kunci)	Tipe
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPerson Affiliation`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPerson OrgUnitDN`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPerson PrimaryAf filiation`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPerson PrincipalName`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPerson Entitlement`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPerson PrimaryOrgUnitDN`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.9`	`eduPerson ScopedAff iliation`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.10`	`eduPerson TargetedID`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.11`	`eduPerson Assurance`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.2.1.2`	`eduOrgHomePageURI`	Daftar string

Atribut eduPerson atau eduOrg (kunci **`Name`**)	Memetakan ke kunci AWS konteks ini (**`FriendlyName`** kunci)	Tipe
`urn:oid:1.3.6.1.4.1.5923.1.2.1.3`	`eduOrgIdentityAuthNPolicyURI`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.2.1.4`	`eduOrgLegalName`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.2.1.5`	`eduOrgSuperiorURI`	Daftar string
`urn:oid:1.3.6.1.4.1.5923.1.2.1.6`	`eduOrgWhitePagesURI`	Daftar string
`urn:oid:2.5.4.3`	`cn`	Daftar string

Tabel berikut menunjukkan atribut Active Directory.

Atribut AD	Memetakan ke kunci AWS konteks ini	Tipe
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	String
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	String

Atribut AD	Memetakan ke kunci AWS konteks ini	Tipe
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	String

Tabel berikut menunjukkan X.500 atribut.

X.500 atribut	Memetakan ke kunci AWS konteks ini	Tipe
`2.5.4.3`	`commonName`	String
`2.5.4.4`	`surname`	String
`2.4.5.42`	`givenName`	String
`2.5.4.45`	`x500UniqueIdentifier`	String
`0.9.2342.19200300100.1.1`	`uid`	String
`0.9.2342.19200300100.1.3`	`mail`	String
`0.9.2342.19200300.100.1.45`	`organizationStatus`	Tali

Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS

Anda dapat menggunakan peran untuk mengonfigurasi penyedia identitas (IDP) yang sesuai dengan SAMP 2.0 Anda dan mengizinkan prinsipal federasi SAMP AWS untuk mengakses file. Konsol Manajemen AWS Peran ini memberikan izin kepada pengguna untuk melaksanakan tugas di konsol. Jika Anda ingin memberi kepala sekolah federasi SAMP cara lain untuk mengakses AWS, lihat salah satu topik berikut:

- AWS CLI: Beralih ke peran IAM (AWS CLI)
- Tools for Windows PowerShell: Beralih ke peran IAM (Alat untuk Windows PowerShell)

- AWS API: Beralih ke peran IAM (AWS API)

Gambaran umum

Diagram berikut menggambarkan aliran untuk sistem masuk SAML-enabled tunggal.

> **Note**
>
> Penggunaan khusus SAMP ini berbeda dari yang lebih umum diilustrasikan di Federasi SAML 2.0 karena alur kerja ini membuka Konsol Manajemen AWS atas nama pengguna. Ini memerlukan penggunaan titik akhir AWS masuk alih-alih langsung memanggil API. `AssumeRoleWithSAML` Titik akhir memanggil API untuk pengguna dan mengembalikan URL yang secara otomatis mengalihkan peramban pengguna ke Konsol Manajemen AWS.

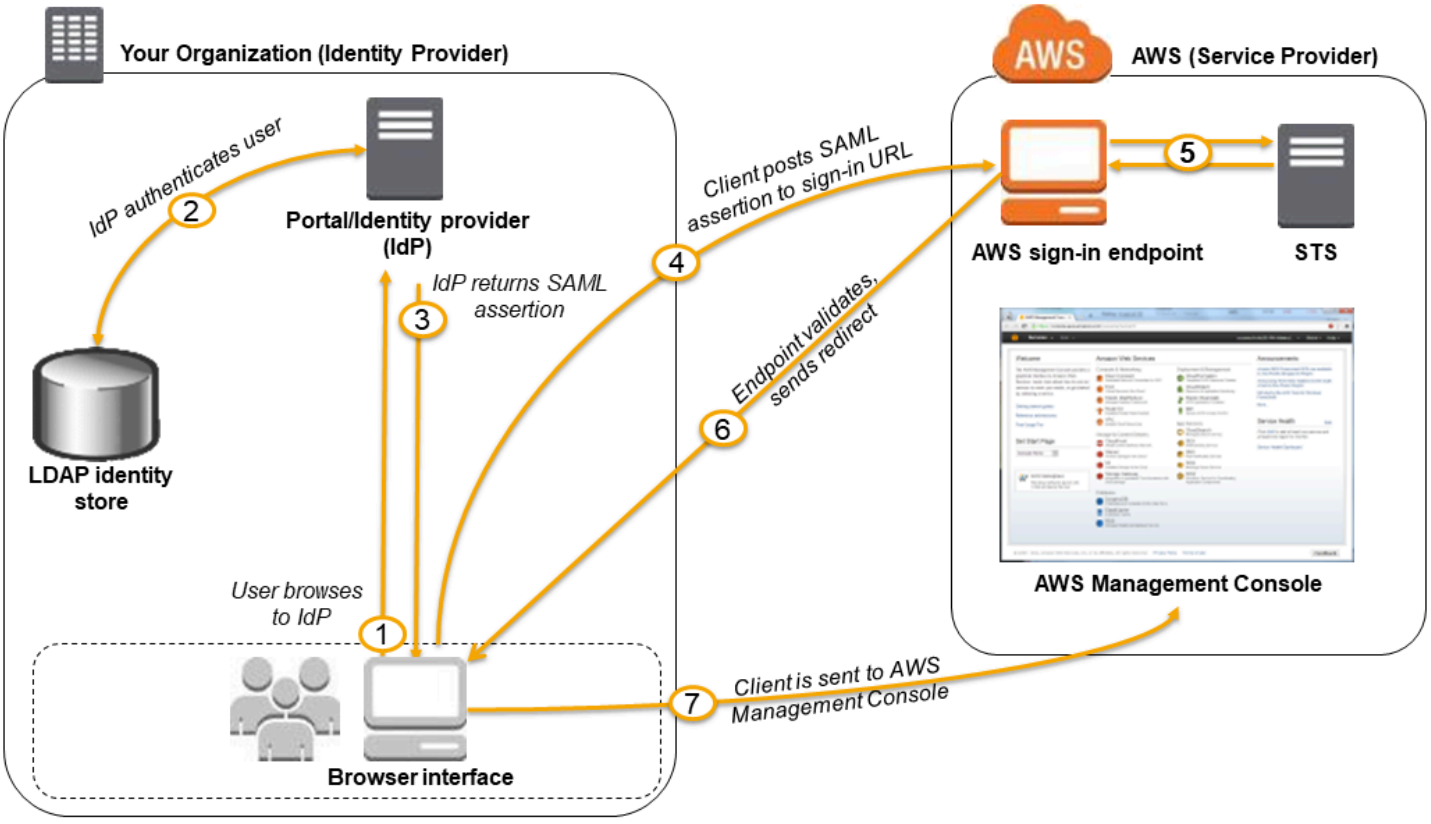


Diagram ini menggambarkan langkah-langkah berikut:

1. Pengguna menelusuri portal organisasi Anda dan memilih opsi untuk membuka Konsol Manajemen AWS. Dalam organisasi Anda, portal biasanya merupakan fungsi dari IDP Anda yang

menangani pertukaran kepercayaan antara organisasi Anda dan organisasi. AWS Misalnya, di Layanan Federasi Active Directory, URL portal adalah: `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`

2. Portal tersebut memverifikasi identitas pengguna di organisasi Anda.
3. Portal tersebut menghasilkan tanggapan autentikasi SAML yang mencakup pernyataan yang mengidentifikasi pengguna dan menyertakan atribut tentang pengguna. Anda juga dapat mengonfigurasi IdP untuk menyertakan atribut pernyataan SAML yang disebut `SessionDuration` yang menentukan durasi sesi konsol yang valid. Anda juga dapat mengonfigurasi IdP untuk meneruskan atribut sebagai tanda sesi. Portal mengirimkan respons ini ke browser klien.
4. Browser klien dialihkan ke titik akhir masuk AWS tunggal dan memposting pernyataan SAMP.
5. Titik akhir meminta kredensial keamanan sementara atas nama pengguna dan membuat URL masuk konsol yang menggunakan kredensial tersebut.
6. AWS mengirim URL masuk kembali ke klien sebagai pengalihan.
7. Peramban klien diarahkan ulang ke Konsol Manajemen AWS. Jika tanggapan autentikasi SAML mencakup atribut yang memetakan ke beberapa peran IAM, pengguna pertama-tama diminta memilih peran untuk mengakses konsol.

Dari sudut pandang pengguna, prosesnya terjadi secara transparan: Pengguna mulai di portal internal organisasi Anda dan berakhir di Konsol Manajemen AWS, tanpa harus memberikan AWS kredensil apa pun.

Lihat bagian berikut untuk gambaran umum tentang cara mengkonfigurasi perilaku ini bersama dengan tautan ke langkah-langkah rinci.

Mengonfigurasi jaringan Anda sebagai penyedia SAML untuk AWS

Di dalam jaringan organisasi Anda, Anda mengonfigurasi toko identitas Anda (seperti Windows Active Directory) untuk bekerja dengan SAML-based iDP seperti Windows Active Directory Federation Services, Shibboleth, dll. Menggunakan IdP, Anda membuat dokumen metadata yang menjelaskan organisasi Anda sebagai IdP dan menyertakan kunci autentikasi. Anda juga mengonfigurasi portal organisasi Anda untuk merutekan permintaan pengguna Konsol Manajemen AWS ke titik akhir AWS SAMP untuk otentikasi menggunakan pernyataan SAMP. Cara Anda mengonfigurasi IdP untuk menghasilkan file metadata.xml tergantung pada IdP Anda. Lihat dokumentasi IdP untuk instruksi, atau lihat Integrasikan penyedia solusi SAML pihak ketiga dengan AWS untuk tautan ke dokumentasi web bagi banyak penyedia SAML yang didukung.

Membuat penyedia SAML di IAM

Selanjutnya, Anda masuk ke Konsol Manajemen AWS dan pergi ke konsol IAM. Di sana Anda membuat penyedia SAMP baru, yang merupakan entitas di IAM yang menyimpan informasi tentang IDP organisasi Anda. Sebagai bagian dari proses ini, Anda mengunggah dokumen metadata yang dibuat oleh perangkat lunak IdP di organisasi Anda pada bagian sebelumnya. Untuk detailnya, lihat Buat penyedia identitas SAML di IAM.

Konfigurasikan izin di AWS untuk kepala sekolah federasi SAMP

Langkah selanjutnya adalah menciptakan peran IAM yang membangun hubungan kepercayaan antara IAM dan IDP organisasi Anda. Peran ini harus mengidentifikasi IdP sebagai prinsipal (entitas tepercaya) untuk tujuan federasi. Peran ini juga menentukan apa yang diizinkan dilakukan oleh pengguna yang diautentikasi oleh IDP organisasi Anda. AWS Anda dapat menggunakan konsol IAM untuk membuat peran ini. Saat Anda membuat kebijakan kepercayaan yang menunjukkan siapa yang dapat mengambil peran tersebut, Anda menentukan penyedia SAML yang Anda buat sebelumnya di IAM. Anda juga menentukan satu atau beberapa atribut SAML yang harus dicocokkan pengguna agar diizinkan untuk menjalankan peran tersebut. Misalnya, Anda dapat menyebutkan bahwa hanya pengguna yang nilai `eduPersonOrgDN` SAML-nya adalah `ExampleOrg` yang diperbolehkan untuk masuk. Wizard peran secara otomatis menambahkan ketentuan untuk menguji atribut `saml:aud` guna memastikan bahwa peran tersebut diambil hanya untuk masuk ke Konsol Manajemen AWS.

Jika enkripsi SAMP diperlukan, URL masuk harus menyertakan pengenal unik yang ditetapkan ke penyedia AWS SAMP Anda, yang dapat Anda temukan di halaman detail penyedia Identitas. Kebijakan kepercayaan untuk peran tersebut mungkin akan terlihat seperti ini:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
```

```
                    "saml:edupersonorgdn": "ExampleOrg",
                    "saml:aud": "https://region-code.signin.aws.amazon.com/saml/
acs/SAMLSP4SHN3UIS2D558H46"
                }
            }
        }
    ]
}
```

> **Note**
>
> IDP SAMP yang digunakan dalam kebijakan kepercayaan peran harus berada dalam akun yang sama dengan peran tersebut.

Sebaiknya gunakan titik akhir regional untuk `saml:aud` atribut di `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Untuk daftar *region-code* nilai yang mungkin, lihat kolom Region di AWS Sign-In titik akhir titik AWS .

Untuk kebijakan izin dalam peran tersebut, Anda menentukan izin seperti yang Anda inginkan untuk peran, pengguna, atau grup. Misalnya, jika pengguna dari organisasi Anda diizinkan untuk mengelola instans Amazon EC2, Anda secara eksplisit mengizinkan tindakan Amazon EC2 dalam kebijakan izin. Anda dapat melakukannya dengan menetapkan kebijakan terkelola, seperti kebijakan terkelola Akses Penuh Amazon EC2.

Untuk detail tentang membuat peran untuk IdP SAML, lihat Buat peran untuk federasi SAML 2.0 (konsol).

Selesaikan konfigurasi dan buat pernyataan SAML

Beri tahu SAMP IDP Anda yang AWS merupakan penyedia layanan Anda dengan menginstal `saml-metadata.xml` file yang ditemukan di atau. `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` `https://signin.aws.amazon.com/static/saml-metadata.xml` Jika enkripsi SAMP diperlukan, file tersebut ditemukan di `https://region-code.signin.aws.amazon.com/static/saml/SAMLSP4SHN3UIS2D558H46/saml-metadata.xml`.

Untuk daftar *region-code* nilai yang mungkin, lihat kolom Region di AWS Sign-In titik akhir titik AWS .

Cara Anda menginstal file tergantung pada IdP Anda. Beberapa penyedia memberi Anda opsi untuk mengetik URL, tempat IdP mendapatkan dan menginstal file untuk Anda. Lainnya mengharuskan Anda mengunduh file dari URL lalu menyediakannya sebagai file lokal. Lihat dokumentasi IdP untuk detail, atau lihat Integrasikan penyedia solusi SAML pihak ketiga dengan AWS untuk tautan ke dokumentasi web bagi banyak penyedia SAML yang didukung.

Anda juga mengonfigurasi informasi yang Anda inginkan untuk diteruskan oleh IdP sebagai atribut SAML ke AWS sebagai bagian dari respons autentikasi. Sebagian besar informasi ini muncul AWS sebagai kunci konteks kondisi yang dapat Anda evaluasi dalam kebijakan Anda. Kunci syarat ini memastikan bahwa hanya pengguna yang diberi otorisasi dalam konteks yang tepat yang diberi izin untuk mengakses sumber daya AWS Anda. Anda dapat menentukan jendela waktu yang dibatasi saat konsol dapat digunakan. Anda juga dapat menentukan waktu maksimum (hingga 12 jam) yang dapat diakses pengguna konsol sebelum harus menyegarkan kembali kredensial mereka. Lihat perinciannya di Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

Melihat respons SAML di browser Anda

Prosedur berikut menjelaskan cara melihat respons SAML dari penyedia layanan Anda dari browser Anda ketika menyelesaikan masalah terkait SAML 2.0.

Untuk semua browser, kunjungi halaman tempat Anda dapat mereproduksi masalah. Kemudian ikuti langkah-langkah untuk peramban yang sesuai:

Topik

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Apa yang harus dilakukan dengan respons Base64-encoded SAML

Google Chrome

Untuk melihat respons SAML di Chrome

Langkah-langkah ini diuji menggunakan versi 106.0.5249.103 (Official Build) (arm64) dari Google Chrome. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tek an F12 untuk memulai konsol Alat Pengembang.

2. Pilih tab Jaringan, lalu pilih Per tahankan log di kiri atas jendela Alat Peng embang.
3. Mereproduksi masalah.
4. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Alat Pengembang, buka menu konteks (klik kanan) untuk label kolom apa pun dan pilih Metode untuk menambahkan kolom.
5. Cari Postingan SAML di panel log Jaringan Alat Pengembang. Pilih baris itu, lalu lihat tab Payload di bagian atas. Cari elemen SamlResponse yang berisi permintaan yang dikodekan. Nilai yang terkait adalah Base64-encoded respons.

Mozilla Firefox

Untuk melihat respons SAML di Firefox

Prosedur ini diuji pada versi 105.0.3 (64-bit) dari Mozilla Firefox. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Tek an F12 untuk memulai konsol Web Developer Tools.
2. Pilih tab Jaringan.
3. Di kanan atas jendela Alat Pengembang Web, pilih opsi (ikon roda gigi kecil). Pilih Pertahankan log.
4. Mereproduksi masalah.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Alat Pengembang Web, buka menu konteks (klik kanan) untuk setiap label kolom dan pilih Metode untuk menambahkan kolom.
6. Cari POST SAML di dalam tabel. Pilih baris itu, lalu lihat tab Per mintaan dan temukan elemen SamlRespon se. Nilai yang terkait adalah Base64-encoded respons.

Apple Safari

Untuk melihat respons SAML di Safari

Langkah-langkah ini diuji menggunakan versi 16.0 (17614.1.25.9.10, 17614) dari Apple Safari. Jika Anda menggunakan versi lain, Anda mungkin perlu mengadaptasi langkah-langkah tersebut.

1. Aktifkan Web Inspector di Safari. Buka jendela Preferensi , pilih tab Lanjutan , lalu pilih Tampilkan menu Kembangkan di bilah menu.

2. Sekarang Anda dapat membuka Web Inspector. Pilih Kembangkan di bilah menu, lalu pilih Show Web Inspector.
3. Pilih tab Jaringan.
4. Di kiri atas jendela Web Inspector, pilih opsi (ikon lingkaran kecil berisi tiga garis horizontal). Pilih Pertahankan Log.
5. (Opsional) Jika kolom Metode tidak terlihat di panel log Jaringan Inspector Web, buka menu konteks (klik kanan) untuk label kolom apa pun dan pilih Metode untuk menambahkan kolom.
6. Mereproduksi masalah.
7. Cari POST SAML di dalam tabel. Pilih baris itu, lalu lihat tab Headers.
8. Cari elemen SamlResponse yang berisi permintaan yang dikodekan. Gulir ke bawah untuk menemukan `Request Data` dengan nama `SAMLResponse`. Nilai yang terkait adalah Base64-encoded respons.

Apa yang harus dilakukan dengan respons Base64-encoded SAML

Setelah Anda menemukan elemen respons Base64-encoded SAML di browser Anda, salin dan gunakan alat Base-64 decoding favorit Anda untuk mengekstrak respons yang ditandai XML.

> **Tips keamanan**
>
> Karena data respons SAML yang Anda lihat mungkin berisi data keamanan yang sensitif, kami menyarankan Anda untuk tidak menggunakan dekoder base64 online. Sebaliknya, gunakan alat yang diinstal di komputer lokal Anda yang tidak mengirimkan data SAML Anda melalui jaringan.

Built-in opsi untuk sistem Windows (PowerShell):

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Built-in opsi untuk sistem macOS dan Linux:

```
$ echo "base64encodedtext" | base64 --decode
```

Tinjau nilai-nilai dalam file yang diterjemahkan

Tinjau nilai-nilai dalam file respons SAML yang didekodekan.

- Verifikasi bahwa nilai untuk atribut SAML:nameid cocok dengan nama pengguna untuk pengguna yang diautentikasi.
- Tinjau nilai untuk Penye https://aws.amazon.com/SAML/Attributes/Role. dia ARN dan SAML peka huruf besar/kecil, dan ARN harus cocok dengan sumber daya di akun Anda.
- Tinjau nilai untuk https://aws.amazon.com/SAML/Attributes/RoleSessionName. Nilai harus sesuai dengan nilai dalam aturan klaim.
- Jika Anda mengonfigurasi nilai atribut untuk alamat email atau nama akun, pastikan nilainya benar. Nilai harus sesuai dengan alamat email atau nama akun pengguna yang diautentikasi.

Periksa kesalahan dan konfirmasikan konfigurasi

Periksa apakah nilai berisi kesalahan, dan konfirmasikan bahwa konfigurasi berikut sudah benar.

- Aturan klaim memenuhi elemen yang diperlukan dan semua ARN benar. Untuk informasi selengkapnya, lihat Konfigurasikan SAML 2.0 IdP Anda dengan kepercayaan pihak yang mengandalkan dan menambahkan klaim.
- Anda mengunggah file metadata terbaru dari IdP Anda ke AWS penyedia SAML Anda. Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS.
- Anda mengonfigurasi kebijakan kepercayaan peran IAM dengan benar. Lihat informasi yang lebih lengkap di Metode untuk mengambil peran.

Federasi AWS Identitas ke layanan eksternal

Federasi identitas keluar IAM memungkinkan AWS beban kerja Anda mengakses layanan eksternal dengan aman tanpa menyimpan kredenSIAL jangka panjang. Beban AWS kerja Anda dapat meminta Token Web JSON (JWT) berumur pendek dari AWS Security Token Service (AWS STS) dengan memanggil API. GetWebIdentityToken Token ini ditandatangani secara kriptografis, dapat diverifikasi secara publik dan berisi serangkaian klaim komprehensif yang menegaskan identitas AWS beban kerja Anda ke layanan eksternal. Anda dapat menggunakan token ini dengan berbagai penyedia cloud pihak ketiga, platform SaaS, dan aplikasi yang di-host sendiri. Layanan eksternal memverifikasi keaslian token menggunakan AWS kunci verifikasi yang diterbitkan di titik akhir terkenal dan menggunakan informasi dalam token untuk membuat keputusan otentikasi dan otorisasi.

Federasi identitas keluar menghilangkan kebutuhan untuk menyimpan kredenSIAL jangka panjang seperti kunci API atau kata sandi dalam kode aplikasi atau variabel lingkungan Anda, sehingga meningkatkan postur keamanan Anda. Anda dapat mengontrol akses ke pembuatan token dan menerapkan properti token seperti algoritma penandatanganan, audiens yang diizinkan, dan durasi menggunakan kebijakan IAM. Semua permintaan token masuk AWS , menyediakan jejak audit lengkap untuk pemantauan keamanan dan pelaporan kepatuhan. Anda juga dapat menyesuaikan token dengan tag yang muncul sebagai klaim khusus, memungkinkan layanan eksternal menerapkan kontrol akses berbasis atribut yang berbutir halus.

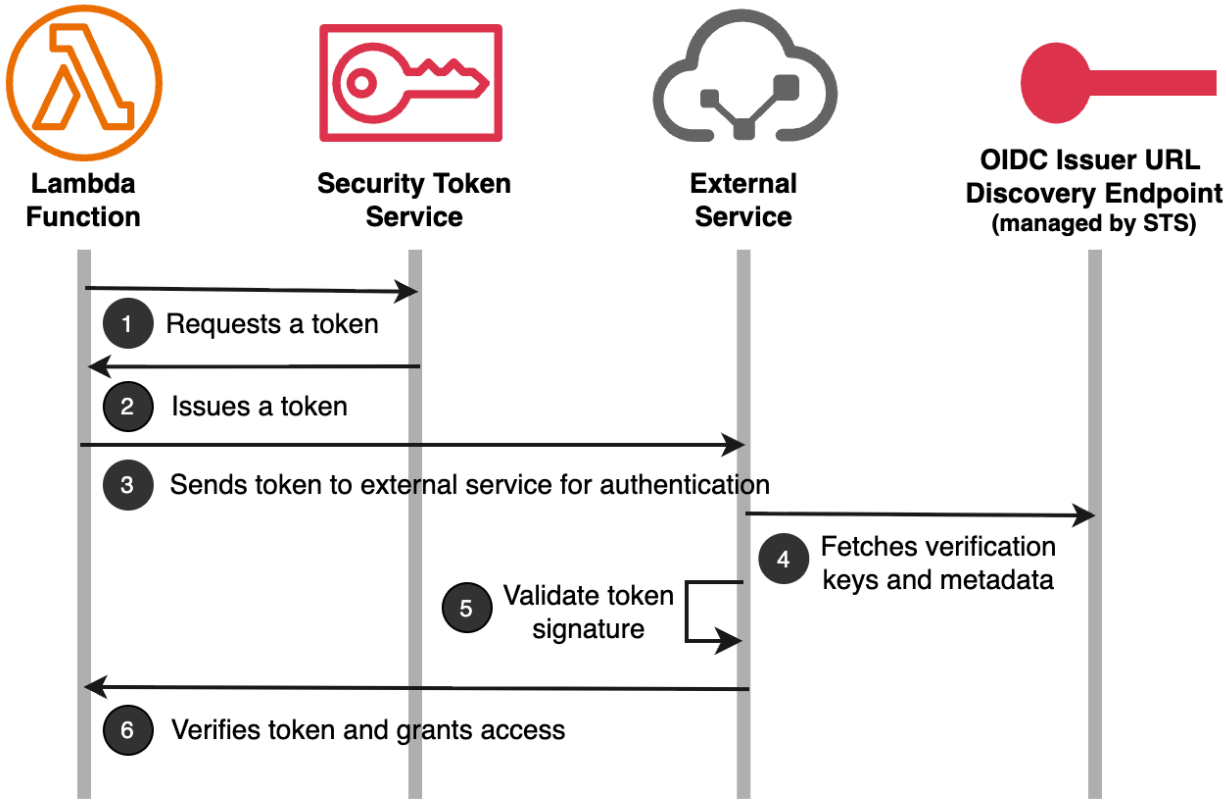
Untuk petunjuk tentang cara mengatur federasi keluar, lihatMemulai dengan federasi identitas keluar. Untuk informasi selengkapnya tentang konten token, lihatMemahami klaim token. Untuk petunjuk tentang mengendalikan beban kerja mana yang dapat meminta token, lihatMengontrol akses dengan kebijakan IAM.

Kasus penggunaan umum

Menggunakan federasi identitas keluar, AWS beban kerja Anda dapat dengan aman:

- Akses sumber daya dan layanan di penyedia cloud eksternal. Misalnya, fungsi Lambda memproses data dapat menulis hasil ke layanan penyimpanan penyedia cloud eksternal atau menanyakan database mereka.
- Integrasikan dengan penyedia perangkat lunak sebagai layanan (SaaS) eksternal untuk analitik, pemrosesan data, pemantauan, dll. Misalnya, fungsi Lambda Anda dapat mengirim metrik ke platform observabilitas.
- Otentikasi dengan aplikasi Anda sendiri yang dihosting di AWS, penyedia cloud eksternal atau pusat data lokal, memungkinkan arsitektur hibrida dan multi-cloud yang aman. Misalnya, AWS beban kerja Anda dapat berinteraksi dengan aplikasi kontainer yang berjalan di cluster Kubernetes lokal Anda.

Cara Kerjanya



1. Fungsi Lambda memanggil GetWebIdentityToken API untuk meminta JSON Web Token (JWT) dari AWS Security Token Service (AWS STS).
2. AWS STS memvalidasi permintaan dan mengembalikan JWT yang ditandatangani ke fungsi Lambda.
3. Fungsi Lambda mengirimkan JWT ke layanan eksternal.
4. Layanan eksternal mengekstrak URL penerbit dari token, memverifikasinya cocok dengan penerbit tepercaya yang diketahui, dan mengambil kunci AWS verifikasi dan metadata dari titik akhir penemuan OIDC.
5. Layanan eksternal menggunakan kunci verifikasi untuk memverifikasi tanda tangan token dan memvalidasi klaim seperti waktu kedaluwarsa, subjek, dan audiens.
6. Setelah validasi berhasil, layanan eksternal memberikan akses ke fungsi Lambda.

Memulai dengan federasi identitas keluar

Panduan ini menunjukkan cara mengaktifkan federasi identitas keluar untuk AWS akun Anda dan mendapatkan Token Web JSON (JWT) pertama Anda (menggunakan GetWebIdentityToken API). Anda akan mengaktifkan fitur, membangun hubungan kepercayaan dengan layanan eksternal, mengonfigurasi izin IAM, dan meminta token menggunakan AWS CLI atau AWS SDK untuk Python (Boto3).

Prasyarat

Sebelum memulai, pastikan Anda memiliki:

- Versi terbaru dari AWS CLI atau Python 3.8 (atau yang lebih baru) dan Boto3 diinstal (untuk AWS contoh SDK)
- Akun layanan eksternal tempat Anda dapat mengonfigurasi hubungan kepercayaan (seperti penyedia cloud eksternal, penyedia SaaS, atau aplikasi pengujian)

> **Note**
>
> - `GetWebIdentityToken`API tidak tersedia di titik akhir STS Global.
> - Token Web JSON (JWT) yang dihasilkan oleh `GetWebIdentityToken` API tidak dapat digunakan untuk federasi OpenID Connect (OIDC) ke AWS (melalui API). AssumeRoleWithWebIdentity

Aktifkan federasi identitas keluar untuk akun Anda

Anda harus mengaktifkan federasi identitas keluar sebelum Anda dapat meminta token. Anda dapat mengaktifkan fitur menggunakan Konsol AWS Manajemen atau secara terprogram menggunakan API. EnableOutboundWebIdentityFederation

Menggunakan AWS CLI

```
aws iam enable-outbound-web-identity-federation
```

Penggunaan AWS SDK for Python

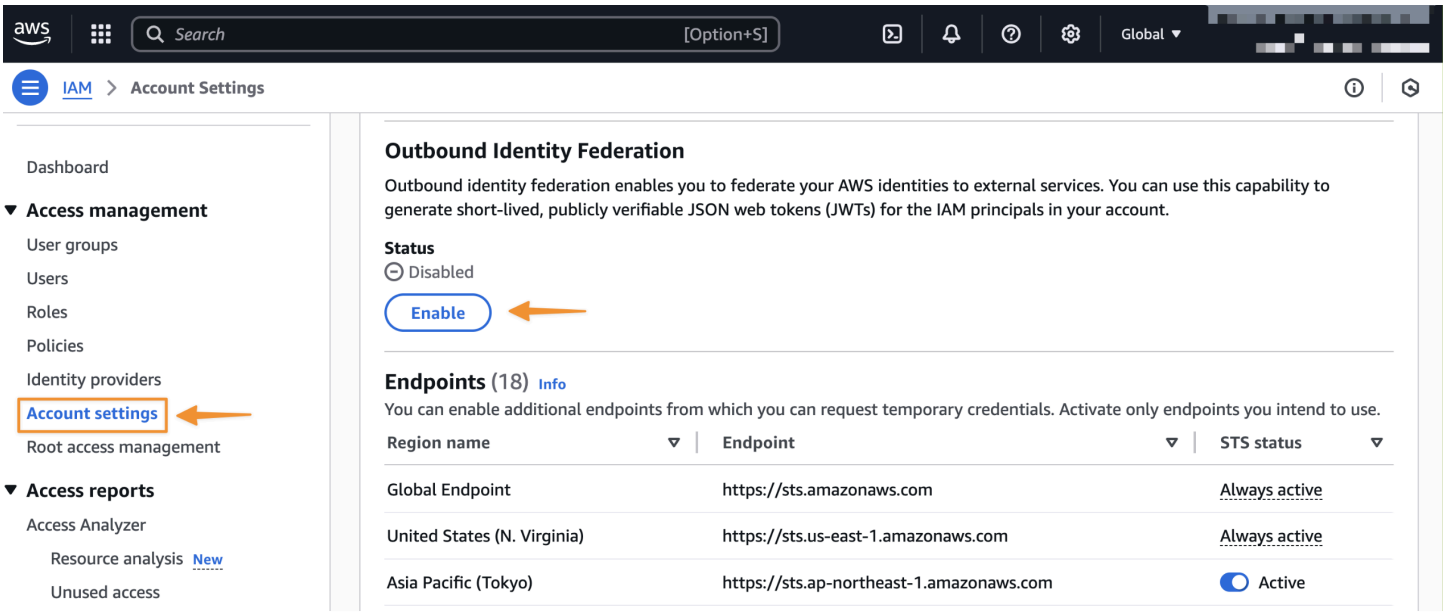
```
import boto3
```

```
# Create IAM client
iam_client = boto3.client('iam')

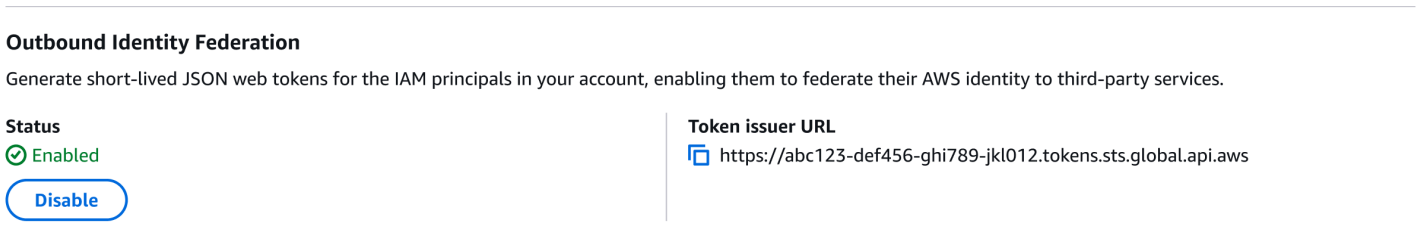
# Enable outbound identity federation
response = iam_client.enable_outbound_web_identity_federation()
print(f"Feature enabled. Issuer URL: {response['IssuerUrl']}")
print(f"Status: {response['Status']}")
```

Menggunakan AWS Konsol

Arahkan ke IAM dan pilih Peng aturan Akun di bawah bagian Manajemen Akses dari menu navigasi sebelah kiri



Setelah Anda mengaktifkan fitur ini, catat URL penerbit khusus akun Anda. Anda akan menggunakan URL ini saat mengonfigurasi hubungan kepercayaan di layanan eksternal. Anda juga dapat mengambil URL penerbit ini sesuai kebutuhan menggunakan GetOutboundWebIdentityFederationInfo API.



Membangun hubungan kepercayaan dalam layanan eksternal

Konfigurasikan layanan eksternal untuk mempercayai dan menerima token yang dikeluarkan oleh AWS akun Anda. Langkah-langkah spesifik bervariasi menurut layanan, tetapi umumnya melibatkan:

- Mendaf AWS tarkan URL penerbit akun Anda sebagai penyedia identitas tepercaya
- Mengkonfigurasi klaim mana yang akan divalidasi (audiens, pola subjek)
- Token pemetaan klaim izin di layanan eksternal

Lihat dokumentasi layanan eksternal untuk petunjuk konfigurasi terperinci.

Mengonfigurasi izin IAM

Buat kebijakan IAM yang memberikan izin untuk memanggil GetWebIdentityToken API dan melampirkan kebijakan ke peran IAM yang perlu menghasilkan token.

Kebijakan contoh ini memberikan akses ke pembuatan token dengan batasan tertentu. Ini memungkinkan permintaan token hanya untuk "https://api.example.com" sebagai audiens dan memberlakukan masa pakai token maksimum 5 menit (300 detik). Lihat daftar kunci kondisi yang dapat Anda gunakan untuk menerapkan properti token. the section called “Kunci kondisi IAM”

Contoh kebijakan IAM

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "sts:IdentityTokenAudience": "https://api.example.com"
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                }
            }
        }
    ]
}
```

Minta Token Web JSON pertama Anda (JWT)

Anda dapat meminta Token Web JSON menggunakan GetWebIdentityToken API. Anda dapat menentukan parameter berikut saat memanggil API:

- Audiens (wajib): Penerima token yang dituju. Nilai ini mengisi klaim “aud” di JWT. Layanan eksternal memvalidasi klaim ini untuk memastikan token ditujukan untuk mereka.
- SigningAlgorithm (wajib): Algorit ma kriptografi yang digunakan untuk menandatangani token. Nilai yang valid adalah ES384 dan RS256. Gunakan ES384 untuk keamanan dan kinerja optimal, atau RS256 untuk kompatibilitas yang lebih luas dengan sistem yang tidak mendukung ECDSA.
- DurationSeconds (opsional): Masa pakai token dalam hitungan detik. Nilai yang valid berkisar dari 60 hingga 3600. Defaultnya adalah 300 (5 menit). Kami merekomendasikan masa pakai token yang lebih pendek untuk meningkatkan keamanan.
- Tag (opsional): Daftar pasangan kunci-nilai untuk disertakan sebagai klaim khusus dalam token. Layanan eksternal dapat menggunakan klaim ini untuk otorisasi berbutir halus.

API mengembalikan bidang berikut:

- IdentityToken: JWT yang ditandatangani sebagai string yang dikodekan base64url. Sertakan token ini dalam permintaan ke layanan eksternal.
- Kedaluwar sa: Stempel waktu UTC untuk saat token kedaluwarsa.

Menggunakan AWS CLI

```
aws sts get-web-identity-token \
    --audience "https://api.example.com" \
    --signing-algorithm ES384 \
    --duration-seconds 300 \
    --tags Key=team,Value=data-engineering \
           Key=environment,Value=production \
           Key=cost-center,Value=analytics
```

Penggunaan AWS SDK for Python

```
import boto3

sts_client = boto3.client('sts')
```

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    DurationSeconds=300,
    SigningAlgorithm='RS256',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'environment', 'Value': 'production'},
        {'Key': 'cost-center', 'Value': 'analytics'}
    ]
)

token = response['WebIdentityToken']
```

Anda juga dapat memecahkan kode JWT untuk memeriksa isinya menggunakan pustaka JWT standar seperti PyJWT, Python-jose untuk Python, Nimbus JOSE+JWT untuk Java atau debugger seperti jwt.io. Lihat the section called “Memahami klaim token” untuk informasi lebih lanjut tentang klaim yang disertakan dalam token.

Gunakan token dengan layanan eksternal

Setelah menerima token, sertakan dalam permintaan ke layanan eksternal. Metode bervariasi menurut layanan, tetapi sebagian besar layanan menerima token di header Otorisasi. Layanan eksternal harus menerapkan logika validasi token yang mengambil kunci JWKS dari titik akhir penerbit Anda yang terkenal, memverifikasi tanda tangan token, dan memvalidasi klaim penting sebelum memberikan akses ke beban kerja Anda. AWS

Ambil kunci verifikasi dan metadata dari titik akhir OpenID Connect (OIDC)

URL penerbit unik untuk AWS akun Anda menghosting titik akhir penemuan OpenID Connect (OIDC) yang berisi kunci verifikasi dan metadata yang diperlukan untuk verifikasi token.

URL titik akhir penemuan OIDC berisi metadata yang digunakan beberapa penyedia untuk memverifikasi token. Ini tersedia di:

```
{issuer_url}/.well-known/openid-configuration
```

Titik akhir JWKS (JSON Web Key Set) berisi kunci yang digunakan untuk memverifikasi tanda tangan token. Ini tersedia di:

```
{issuer_url}/.well-known/jwks.json
```

Ambil JWKS menggunakan curl

```
curl https://{issuer_url}/.well-known/jwks.json
```

Respons:

```
{
  "keys": [
    {
      "kty": "EC",
      "use": "sig",
      "kid": "key-id-1",
      "alg": "ES384",
      "crv": "P-384",
      "x": "base64-encoded-x-coordinate",
      "y": "base64-encoded-y-coordinate"
    },
    {
      "kty": "RSA",
      "use": "sig",
      "kid": "key-id-2",
      "n": "base64-encoded-modulus",
      "e": "AQAB"
    }
  ]
}
```

Penggunaan AWS SDK for Python

```
import requests

# Fetch Openid Configuration
open_id_config_response = requests.get("https://{issuer_url}/.well-known/openid-
configuration")
open_id_config = open_id_config_response.json()

# Fetch JWKS
jwks_response = requests.get("https://{issuer_url}/.well-known/jwks.json")
jwks = jwks_response.json()
```

Kami menyarankan untuk menyimpan kunci ini dalam cache untuk menghindari pengambilan kunci tersebut untuk setiap verifikasi token.

Validasi klaim penting

- Subjek (sub): Verifikasi klaim subjek berisi pola ARN utama IAM yang diharapkan.
- Kedaluwarsa (exp): Pastikan token belum kedaluwarsa. Pustaka JWT biasanya menangani ini secara otomatis.
- Audiens (aud): Pastikan audiens cocok dengan nilai yang Anda harapkan. Ini mencegah token yang ditujukan untuk layanan lain digunakan bersama Anda.
- Penerbit (iss): Verifikasi penerbit cocok AWS dengan akun yang Anda percayai. Simpan daftar URL penerbit tepercaya.

Bila memungkinkan, Anda harus memvalidasi AWS klaim spesifik tambahan untuk menerapkan kontrol akses halus di layanan eksternal Anda. Misalnya, validasi klaim org_id untuk membatasi akses ke prinsipal IAM di AWS Organisasi Anda, periksa principal_tags untuk menerapkan kontrol akses berbasis atribut (seperti mengizinkan hanya lingkungan produksi atau tim tertentu), atau verifikasi klaim konteks sesi seperti lambda_source_function_arn atau ec2_instance_source_vpc untuk membatasi akses berdasarkan sumber daya komputasi. Lihat Mem ahami klaim token untuk daftar lengkap klaim yang disertakan dalam token.

Memahami klaim token

Saat Anda memanggil GetWebIdentityToken API, Layanan Token AWS Keamanan mengembalikan Token Web JSON (JWT) yang ditandatangani yang berisi serangkaian klaim yang mewakili identitas prinsipal IAM. Token ini sesuai dengan RFC 75 19. Memahami struktur dan isi token ini membantu Anda menerapkan alur otentikasi aman, mengonfigurasi validasi klaim yang sesuai di layanan eksternal, dan secara efektif menggunakan klaim khusus untuk kontrol akses yang halus.

JWT mencakup klaim OpenID Connect (OIDC) standar seperti subjek (“sub”), audiens (“aud”), penerbit (“iss”) untuk memfasilitasi interoperabilitas di berbagai layanan eksternal. AWS STS mengisi token dengan klaim AWS khusus identitas (seperti ID AWS Akun dan tag Principal) dan klaim konteks sesi (seperti ARN instans EC2) bila berlaku. Anda juga dapat menambahkan klaim khusus ke token dengan meneruskannya sebagai tag permintaan ke GetWebIdentityToken API. K AWS laim khusus identitas, dan klaim konteks sesi dan klaim khusus bersarang di bawah namespace "https://sts.amazonaws.com/" di token.

Lihat token sampel di bawah ini untuk daftar klaim yang disertakan dalam token. Harap dicatat bahwa semua klaim ini mungkin tidak hadir dalam token pada saat yang bersamaan.

```
{
```

```
  "iss": "https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws",
  "aud": "https://api.example.com",
  "sub": "arn:aws:iam::123456789012:role/DataProcessingRole",
  "iat": 1700000000,
  "exp": 1700000900,
  "jti": "xyz123-def456-ghi789-jkl012",
  "https://sts.amazonaws.com/": {
    "aws_account": "123456789012",
    "source_region": "us-east-1",
    "org_id": "o-abc1234567",
    "ou_path": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/",
    "principal_tags": {
      "environment": "production",
      "team": "data-engineering",
      "cost-center": "engineering"
    },
    "lambda_source_function_arn": "arn:aws:lambda:us-
east-1:123456789012:function:process-data",
    "request_tags": {
        "job-id": "job-2024-001",
        "priority": "high",
        "data-classification": "sensitive"
    }
  }
}
```

Klaim standar

Klaim OIDC standar yang ada dalam token memfasilitasi interoperabilitas dengan berbagai layanan eksternal. Klaim ini dapat divalidasi menggunakan sebagian besar pustaka JWT.

Klaim	Nama	Deskripsi	Nilai Contoh
is	Penerbit	URL penerbit khusus akun Anda. Layanan eksternal memvalidasi klaim ini untuk memastikannya cocok dengan penerbit tepercaya mereka.	https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws

Klaim	Nama	Deskripsi	Nilai Contoh
aud	Audiens	Penerima yang dituju untuk token yang ditentukan dalam GetWebIdentityToken permintaan.	https://api.example.com
sub	Subjek	ARN dari kepala sekolah IAM yang meminta token.	Arn:aws:iam: :123456789012:role/DataProcessingRole
is	Diterbitkan Pada	NumericDate nilai yang mengidentifikasi waktu di mana JWT dikeluarkan.	1700000000
exp	Kedaluwarsa	NumericDate nilai yang mengidentifikasi waktu kedaluwarsa pada atau setelah itu JWT TIDAK HARUS diterima untuk diproses.	1700000900
jti	JWT ID	Pengidentifikasi unik untuk contoh token ini.	xyz123-def456-ghi789-jkl012

Klaim khusus

Selain klaim OIDC standar, AWS STS menambahkan klaim tentang identitas dan konteks sesi bila berlaku. Anda juga dapat menambahkan klaim Anda sendiri ke token dengan meneruskannya sebagai tag permintaan. Klaim khusus bersarang di bawah https://sts.amazonaws.com/ namespace.

AWS klaim identitas

Klaim ini memberikan informasi terperinci tentang AWS akun, struktur organisasi, dan prinsipal IAM Anda.

Klaim	Deskripsi	Peta ke Condition Key	Nilai Contoh
aws_account	ID AWS akun Anda	aw: Principal Account	123456789012
sumber_wilayah	Wil AWS ayah tempat token diminta	aw: Requested Region	us-east-1
org_id	ID AWS Organisasi Anda (jika akun Anda adalah bagian dari organisasi)	aws: PrincipalOrg ID	o-abc1234567
ou_path	Jalur unit organisasi Anda (jika ada)	aw: Principal OrgPaths	o- a1b2c3d4e 5/r - ab12/ou - ab12- -ab12-222 22222/11111111/ ou
tag_utama	Tag yang dilampirkan ke sesi peran utama atau asumsi IAM. Ketika token diminta di mana prinsipal IAM yang meminta memiliki tag utama dan tag sesi, tag sesi akan ada di JWT.	aw:PrincipalTag/<tag-key>	{"environment": “production”, “team”: “rekayasa data”, “pusat biaya” :"rekayasa "}

Klaim konteks sesi

Klaim ini memberikan informasi tentang lingkungan komputasi dan sesi tempat permintaan token berasal. AWS STS secara otomatis memasukkan klaim ini bila berlaku berdasarkan konteks sesi kepala sekolah yang meminta.

Klaim	Deskripsi	Peta ke Condition Key	Nilai Contoh
asli_session_exp	Kapan kredenSIAL sesi peran asli akan kedaluwarsa (untuk peran yang diasumsikan)	N/A	2024-01-15 T10:00:00Z
penyedia federasi	Nama penyedia identitas untuk sesi federasi	aw: Federated Provider	arn:aws:i am: :11112222 3333:oidc-provider/your_oidc _provider
identity_store_use r_id	ID pengguna Pusat Identitas IAM	toko identitas: UserId	pengguna-abc123def456
identity_store_arn	ARN dari toko identitas Pusat Identitas	toko identitas: IdentityStoreArn	arn: aws:ident itystore: :12345678 9012: -abc1234567 identitystore/d
ec2_sourc e_instance_arn	ARN instans EC2 yang meminta	ec2: SourceIns tanceArn	arn: aws:ec2: us-timur- 1:1234567 89012: -abc123de f456 instance/i
ec2_insta nce_source_vpc	ID VPC tempat kredentif peran EC2 dikirimkan	aws: Ec2Instan ceSourceVpc	vpc-abc123def456
ec2_insta nce_source_private _ipv4	Alamat IPv4 pribadi dari instance EC2	aws: Ec2Instan ceSourceP rivateIPv4	10.0.1.25
ec2_role_delivery	Versi layanan metadata instans	ec2: RoleDelivery	2
identitas_sumber	Identitas sumber ditetapkan oleh kepala sekolah	aws: SourceIde ntity	admin-pengguna

Klaim	Deskripsi	Peta ke Condition Key	Nilai Contoh
lambda_source_fungsi_arn	ARN dari fungsi memanggil Lambda	Lambda: SourceFunctionArn	arn: aws:lambda: us-east- 1:123456789012: fungsi: fungsi saya
layanan_penerbitan_kredensi_glue_credential_	AWS Pengidentifikasi layanan lem untuk pekerjaan Lem	lem: CredentialIssuingService	perekat.amazonaws.com

Minta tag

Anda dapat menambahkan klaim khusus ke token dengan menentukan tag dalam permintaan GetWebIdentityToken API. Klaim ini muncul di bawah bidang request_tags di token dan memungkinkan Anda untuk menyampaikan informasi spesifik yang dapat digunakan layanan eksternal untuk keputusan otorisasi yang halus. Anda dapat menentukan hingga 50 tag per permintaan.

Contoh permintaan:

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    SigningAlgorithm='ES384',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'cost-center', 'Value': 'analytics'},
        {'Key': 'environment', 'Value': 'production'}
    ]
)
```

Klaim yang dihasilkan dalam token:

```
{
  "request_tags": {
    "team": "data-engineering",
    "cost-center": "analytics",
    "environment": "production"
```

```
  }
}
```

Mengontrol akses dengan kebijakan IAM

IAM menyediakan beberapa jenis kebijakan untuk mengontrol akses ke fitur federasi identitas keluar. Anda dapat menggunakan kebijakan berbasis identitas untuk mengontrol prinsipal IAM mana yang dapat meminta token dan menerapkan properti token tertentu seperti audiens, masa pakai, dan algoritma penandatanganan. Kebijakan Kontrol Layanan (SCP) memungkinkan Anda menerapkan pembatasan seluruh organisasi pada pembuatan token di semua akun di Organisasi Anda. AWS Kebijakan Kontrol Sumber Daya (RCP) mengontrol akses di tingkat sumber daya. Anda juga dapat menggunakan kebijakan titik akhir VPC untuk membatasi prinsipal mana yang dapat mengakses AWS STS `GetWebIdentityToken` API melalui titik akhir VPC Anda, menambahkan kontrol tingkat jaringan ke postur keamanan Anda. Bagian ini menjelaskan cara menerapkan kontrol akses halus menggunakan jenis kebijakan dan kunci kondisi ini.

Untuk meminta token identitas, kepala sekolah IAM harus memiliki `sts:GetWebIdentityToken` izin. Berikan izin ini melalui kebijakan identitas yang dilampirkan ke pengguna atau peran IAM. Untuk mengizinkan Tag (kunci, pasangan nilai) diteruskan ke GetWebIdentityToken panggilan, prinsipal IAM harus memiliki `sts:TagGetWebIdentityToken` izin.

- Gunakan kunci IdentityTokenAudience syarat sts: untuk membatasi layanan eksternal mana yang dapat menerima token.
- Gunakan sts: DurationSeconds condition key untuk menegakkan masa pakai token maksimum.
- Gunakan kunci SigningAlgorithm kondisi sts: untuk memerlukan algoritma kriptografi tertentu.
- Gunakan kunci RequestTag kondisi aws: bandingkan pasangan kunci-nilai tag yang diteruskan dalam permintaan dengan pasangan tag yang Anda tentukan dalam kebijakan.
- Gunakan kunci TagKeys kondisi aws: untuk membandingkan kunci tag dalam permintaan dengan kunci yang Anda tentukan dalam kebijakan.

Lihat IAM dan kunci AWS STS kondisi untuk mempelajari lebih lanjut tentang kunci kondisi yang tersedia di kebijakan IAM.

Kebijakan identitas sampel ini menggabungkan beberapa kunci kondisi:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowTokenGenerationWithRestrictions",
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api1.example.com",
                        "https://api2.example.com"
                    ]
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                },
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

Praktik terbaik

Ikuti rekomendasi ini untuk menyatukan AWS identitas Anda dengan aman ke layanan eksternal.

- Gunakan masa pakai token pendek: Minta token dengan masa pakai terpendek yang memenuhi kebutuhan operasional Anda.
- Menerapkan akses hak istimewa terkecil dan batasi properti token dengan kebijakan IAM: Ber ikan `sts:GetWebIdentityToken` izin hanya kepada prinsipal IAM yang memerlukannya. Gunakan kunci kondisi untuk menentukan algoritma penandatanganan, pemirsa token yang diizinkan, dan masa pakai token maksimum sesuai kebutuhan Anda.
- Validasi klaim dalam layanan eksternal: Untuk keamanan, selalu validasi klaim yang relevan seperti subjek (“sub”), audiens (“aud”), dll. untuk memastikan klaim tersebut sesuai dengan nilai yang Anda harapkan. Validasi klaim kustom bila memungkinkan untuk mengaktifkan keputusan otorisasi berbutir halus di layanan eksternal.

Kredensial keamanan sementara di IAM

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan memberikan kredentif keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke AWS sumber daya Anda. KredenSIAL keamanan sementara bekerja hampir identik dengan kredenSIAL kunci akses jangka panjang, dengan perbedaan berikut:

- Kredensial keamanan sementara bersifat jangka pendek, seperti namanya. Konfigurasi dapat berlangsung selama beberapa menit hingga beberapa jam. Setelah kredenSIAL kedaluwarsa, AWS tidak lagi mengenalinya atau mengizinkan segala jenis akses dari permintaan API yang dibuat dengannya.
- Kredensial keamanan sementara tidak disimpan dengan pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. Ketika (atau bahkan sebelum) kredensial keamanan sementara kedaluwarsa, pengguna dapat meminta kredensial baru, selama pengguna yang memintanya masih memiliki izin untuk melakukannya.

Akibatnya, kredenSIAL sementara memiliki keunggulan berikut dibandingkan kredenSIAL jangka panjang:

- Anda tidak perlu mendistribusikan atau menyematkan kreden AWS tif keamanan jangka panjang dengan aplikasi.
- Anda dapat memberikan akses ke AWS sumber daya Anda kepada pengguna tanpa harus menentukan AWS identitas untuk mereka. KredenSIAL sementara adalah dasar untuk peran dan federasi identitas.
- KredenSIAL keamanan sementara memiliki masa pakai yang terbatas, jadi Anda tidak perlu memperbaruinya atau mencabutnya secara eksplisit ketika tidak lagi diperlukan. Setelah kredensial keamanan sementara berakhir, kredensial tersebut tidak dapat digunakan kembali. Anda dapat menentukan berapa lama kredensial berlaku, hingga batas maksimum.

AWS STS and AWS daerah

Kredensial keamanan sementara dihasilkan oleh AWS STS. Secara default, AWS STS adalah layanan global dengan titik akhir tunggal di`https://sts.amazonaws.com`. Namun, Anda juga dapat memilih untuk melakukan panggilan AWS STS API ke titik akhir di Wilayah lain yang didukung. Ini dapat mengurangi latensi (server lag) dengan mengirim permintaan ke server di Wilayah yang secara geografis lebih dekat dengan Anda. Tidak peduli dari Wilayah mana kredensial Anda berasal,

mereka bekerja secara global. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Skenario umum untuk kredensial sementara

Kredensial sementara berguna dalam skenario yang melibatkan federasi identitas, pendelegasian, akses lintas akun, dan peran IAM.

Federasi identitas

Anda dapat mengelola identitas pengguna di sistem eksternal di luar AWS dan memberi pengguna yang masuk dari sistem tersebut akses untuk melakukan AWS tugas dan mengakses AWS sumber daya Anda. IAM mendukung dua jenis federasi identitas. Dalam kedua kasus, identitas disimpan di luar AWS. Perbedaannya adalah tempat sistem eksternal berada—di pusat data Anda atau pihak ketiga eksternal di web. Untuk membandingkan fitur kredentif keamanan sementara untuk federasi identitas, lihatBandingkan AWS STS kredensialnya.

Untuk informasi lebih lanjut tentang penyedia identitas eksternal, lihat Penyedia identitas dan federasi ke AWS.

- Federasi OpenID Connect (OIDC) - Anda dapat mengizinkan pengguna masuk menggunakan penyedia identitas pihak ketiga yang terkenal seperti Login dengan Amazon, Facebook, Google, atau penyedia apa pun OIDC-compatible . Untuk aplikasi seluler atau web Anda, Anda tidak perlu membuat kode masuk khusus atau mengelola identitas pengguna Anda sendiri. Menggunakan federasi OIDC membantu Anda menjaga keamanan Akun AWS , karena Anda tidak perlu mendistribusikan kredentif keamanan jangka panjang, seperti kunci akses pengguna IAM, dengan aplikasi Anda. Untuk informasi selengkapnya, lihat Federasi OIDC.

  AWS STS Federasi OIDC mendukung Login dengan Amazon, Facebook, Google, dan penyedia identitas yang kompatibel dengan OpenID Connect (OIDC).

  > **Note**
  >
  > Untuk aplikasi seluler, kami sarankan Anda menggunakan Amazon Cognito. Anda dapat menggunakan layanan ini dengan AWS SDK untuk pengembangan seluler untuk membuat identitas unik bagi pengguna dan mengotentikasi mereka untuk akses aman ke sumber daya Anda AWS . Amazon Cognito mendukung penyedia identitas yang sama dengan AWS STS, dan juga mendukung akses yang tidak diautentikasi (tamu) dan memungkinkan Anda memigrasikan data pengguna saat pengguna masuk. Amazon Cognito juga

menyediakan operasi API untuk mensinkronisasi data pengguna sehingga data terjaga saat pengguna berpindah antar perangkat. Untuk informasi selengkapnya, lihat Oten tikasi dengan Amplify di Dokumentasi Amplify.

- Federasi SAML — Anda dapat mengotentikasi pengguna di jaringan organisasi Anda, dan kemudian memberikan akses kepada pengguna tersebut AWS tanpa membuat AWS identitas baru untuk mereka dan mengharuskan mereka untuk masuk dengan kredentif masuk yang berbeda. Ini dikenal sebagai pendekatan sign-on tunggal untuk akses sementara. AWS STS mendukung standar terbuka seperti Security Assertion Markup Language (SAML) 2.0, yang dengannya Anda dapat menggunakan Microsoft AD FS untuk memanfaatkan Microsoft Active Directory Anda. Anda juga dapat menggunakan SAML 2.0 untuk mengelola solusi Anda sendiri untuk memfederasi identitas pengguna. Untuk informasi selengkapnya, lihat Federasi SAML 2.0.
  - Broker federasi kustom — Anda dapat menggunakan sistem otentikasi organisasi Anda untuk memberikan akses ke AWS sumber daya. Untuk contoh skenario, lihat Aktifkan akses broker identitas khusus ke AWS konsol.
  - Federasi menggunakan SAML 2.0 — Anda dapat menggunakan sistem otentikasi organisasi Anda dan SAML untuk memberikan akses ke AWS sumber daya. Untuk informasi lebih lanjut dan contoh skenario, lihat Federasi SAML 2.0.

Peran untuk akses lintas akun

Banyak organisasi mempertahankan lebih dari satu Akun AWS. Dengan menggunakan peran dan akses lintas akun, Anda dapat menentukan identitas pengguna dalam satu akun, dan menggunakan identitas tersebut untuk mengakses sumber daya AWS di akun lain yang menjadi bagian dari organisasi Anda. Ini dikenal sebagai pendekatan pendelegasian ke akses sementara. Untuk informasi selengkapnya tentang cara membuat peran lintas akun, lihat Buat peran untuk memberikan izin kepada pengguna IAM. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengasumsikan peran Anda, lihat Apa yang dimaksud dengan IAM Access Analyzer?.

Peran untuk Amazon EC2

Jika Anda menjalankan aplikasi di instans Amazon EC2 dan aplikasi tersebut memerlukan akses ke AWS sumber daya, Anda dapat memberikan kredentif keamanan sementara ke instans Anda saat meluncurkannya. Kredensial keamanan sementara ini tersedia untuk semua aplikasi yang berjalan pada instans, sehingga Anda tidak perlu menyimpan kredensial jangka panjang apa pun

pada instans. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2.

Untuk mempelajari lebih lanjut tentang kredentif peran IAM Amazon EC2, lihat peran IAM untuk Amazon EC2 di Panduan Pengguna Amazon Elastic Compute Cloud.

Lainnya AWS layanan

Anda dapat menggunakan kredentif keamanan sementara untuk mengakses sebagian besar AWS layanan. Untuk daftar layanan yang menerima kredensial keamanan sementara, lihat AWS layanan yang bekerja dengan IAM.

Contoh aplikasi yang menggunakan kredensial sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan memberikan kredentif keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke AWS sumber daya Anda. Untuk informasi lebih lanjut tentang AWS STS, lihatKredensial keamanan sementara di IAM. Untuk melihat bagaimana Anda dapat menggunakan AWS STS untuk mengelola kredenSIAL keamanan sementara, Anda dapat mengunduh contoh aplikasi berikut yang menerapkan skenario contoh lengkap:

- Mengaktifkan Federasi untuk AWS Menggunakan Windows Active Directory, ADFS, dan SAML 2.0. Menunjukkan cara mendelegasikan akses menggunakan federasi perusahaan untuk AWS menggunakan Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0, dan SAML (Security Assertion Markup Language) 2.0.
- Aktifkan akses broker identitas khusus ke AWS konsol. Menunjukkan cara membuat proxy federasi kustom yang mengaktifkan single sign-on (SSO) sehingga pengguna Active Directory yang ada dapat masuk ke. Konsol Manajemen AWS
- Cara Menggunakan Shibboleth untuk Single Sign-On to the. Konsol Manajemen AWS. Menunjukkan cara menggunakan Shibboleth dan SAML untuk memberikan akses pengguna dengan sistem masuk tunggal (SSO) ke Konsol Manajemen AWS.

Sampel untuk federasi OIDC

Contoh aplikasi berikut menggambarkan cara menggunakan federasi OIDC dengan penyedia seperti Login dengan Amazon, Amazon Cognito, Facebook, atau Google. Anda dapat menukar otentikasi dari penyedia ini untuk kreden AWS SIAL keamanan sementara untuk mengakses AWS layanan.

- Tutorial Amazon Cognito — Kami menyarankan Anda menggunakan Amazon Cognito dengan AWS SDK untuk pengembangan seluler. Amazon Cognito adalah cara paling mudah untuk mengelola identitas untuk aplikasi seluler, serta menyediakan fitur tambahan seperti sinkronisasi dan identitas lintas perangkat. Untuk informasi selengkapnya tentang Amazon Cognito, lihat Oten tikasi dengan Amplify di Dokumentasi Amplify.

Sumber daya tambahan untuk kredensial keamanan sementara

Skenario dan aplikasi berikut dapat memandu Anda dalam menggunakan kredensial keamanan sementara:

- Cara berintegrasi AWS STS SourceIdentity dengan penyedia identitas Anda. Posting ini menunjukkan cara mengatur AWS STS `SourceIdentity` atribut saat menggunakan Okta, Ping, atau OneLogin sebagai IdP Anda.
- Federasi OIDC. Bagian ini membahas cara mengonfigurasi peran IAM saat Anda menggunakan federasi OIDC dan API. `AssumeRoleWithWebIdentity`
- Akses API aman dengan MFA. Topik ini menjelaskan cara menggunakan peran untuk mewajibkan Autentikasi Multi-Faktor (MFA) untuk melindungi tindakan API rahasia di akun Anda.

Untuk informasi selengkapnya tentang kebijakan dan izin di AWS lihat topik berikut:

- Manajemen akses untuk AWS sumber daya
- Logika evaluasi kebijakan.
- Mengelola Izin Akses ke Sumber Daya Amazon S3 Anda di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.
- Untuk mempelajari apakah prinsip di akun di luar zona kepercayaan (organisasi atau akun tepercaya) memiliki akses untuk menjalankan peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

Bandingkan AWS STS kredensialnya

Tabel berikut membandingkan fitur operasi API AWS STS yang mengembalikan kredensil keamanan sementara tersebut. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk meminta kredensial keamanan sementara dengan mengambil peran, lihat Metode untuk mengambil

peran. Untuk mempelajari berbagai operasi AWS STS API yang memungkinkan Anda meneruskan tag sesi, lihatLewati tag sesi di AWS STS.

> **Note**
>
> Anda dapat mengirim panggilan AWS STS API ke titik akhir global atau ke salah satu titik akhir Regional. Jika Anda memilih titik akhir yang lebih dekat dengan Anda, Anda dapat mengurangi latensi dan meningkatkan kinerja panggilan API Anda. Anda juga dapat memilih untuk mengarahkan panggilan Anda ke titik akhir Regional alternatif jika Anda tidak lagi dapat berkomunikasi dengan titik akhir awal. Jika Anda menggunakan salah satu dari beragam AWS SDKs, gunakan metode SDK tersebut untuk menentukan Region sebelum Anda melakukan panggilan API. Jika Anda membuat permintaan HTTP API secara manual, maka Anda harus mengarahkan permintaan ke titik akhir yang benar sendiri. Untuk informasi lebih lanjut, lihat bagian AWS STS dari Wilayah dan Titik Akhir dan Kelola AWS STS dalam sebuah Wilayah AWS.

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	Support MFA[1]	Dukungan kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
AssumeRe	Pengguna IAM atau peran IAM dengan kredensial keamanan sementara yang ada	15 m | Pengaturan durasi sesi maksimum | 1 jam	Ya	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .
AssumeReWithSAM	Setiap pengguna; penelepon harus melewati tanggapan autentikasi SAML yang menunjukk	15 m | Pengaturan durasi sesi maksimum | 1 jam	Tidak	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	Support MFA[1]	Dukungan kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
	an autentikasi dari penyedia layanan yang dikenal				
AssumeRoleWithWebIdentity	Setiap pengguna; penelepon harus melewati token JWT yang sesuai dengan OIDC yang menunjukkan otentikasi dari penyedia identitas yang dikenal	15 m | Pengaturan durasi sesi maksimum | 1 jam	Tidak	Ya	Tidak bisa menelepon `GetFederationToken` atau `GetSessionToken` .
GetFederationToken	Pengguna IAM atau Pengguna root akun AWS	Pengguna IAM: 15 m | 1 jam | 1 jam Pengguna root: 15 m | 1 jam | 1 jam	Tidak	Ya	Tidak dapat memanggil operasi IAM menggunakan AWS CLI atau AWS API. Batasan ini tidak berlaku untuk sesi konsol. Tidak dapat memanggil AWS STS operasi kecuali `GetCallerIdentity` .1 SSO ke konsol diperbolehkan.[5]

AWS STS API	Siapa yang bisa menelepon	Masa pakai kredensi (min | maks | default)	Support MFA[1]	Dukungan kebijakan sesi[2]	Pembatasan pada kredensi sementara yang dihasilkan
GetSessionToken	Pengguna IAM atau Pengguna root akun AWS	Pengguna IAM: 15 m | 1 jam | 1 jam Pengguna root: 15 m | 1 jam | 1 jam	Ya	Tidak	Tidak dapat memanggil operasi API IAM kecuali informasi MFA disertakan dengan permintaan. Tidak dapat memanggil operasi AWS STS API kecuali `AssumeRole` atau`GetCallerIdentity` . SSO ke konsol tidak diperbolehkan.[6]

[1] Support MFA Anda dapat menyertakan informasi tentang perangkat otentikasi multi-faktor (MFA) saat Anda memanggil operasi dan API. AssumeRole GetSessionToken Hal ini memastikan kredensial keamanan sementara yang dihasilkan dari panggilan API hanya dapat digunakan oleh pengguna yang diautentikasi dengan perangkat MFA. Untuk informasi selengkapnya, lihat Akses API aman dengan MFA.

[2] Dukungan kebijakan sesi. Kebijakan sesi adalah kebijakan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau sesi pengguna AWS STS gabungan. Kebijakan ini membatasi izin dari peran atau kebijakan berbasis identitas pengguna yang ditetapkan untuk sesi. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas entitas dan kebijakan sesi. Kebijakan sesi tidak dapat digunakan untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang diasumsikan. Untuk informasi lebih lanjut tentang izin sesi peran, lihat Kebijakan sesi.

[3] Pengaturan durasi sesi maksimum. Gunakan parameter `DurationSeconds` untuk menentukan durasi sesi peran Anda dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk

peran tersebut. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

GetCallerIdentity. Tidak ada izin yang diperlukan untuk melakukan operasi ini. Jika administrator menambahkan kebijakan ke pengguna IAM Anda atau peran yang secara jelas menghalangi akses ke tindakan `sts:GetCallerIdentity`, Anda masih dapat melakukan operasi ini. Izin tidak diperlukan karena informasi yang sama dikembalikan saat pengguna IAM atau peran ditolak aksesnya. Untuk melihat contoh tanggapan, lihat Saya tidak berwenang untuk melakukan: iam:DeleteVirtualMFADevice.

[5] Masuk tunggal (SSO) ke konsol. Untuk mendukung SSO, Anda AWS dapat memanggil federasi endpoint (`https://signin.aws.amazon.com/federation`) dan meneruskan kredensi keamanan sementara. Titik akhir mengembalikan token yang dapat Anda gunakan untuk membangun URL yang menandai pengguna secara langsung ke konsol tanpa memerlukan kata sandi. Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS dan Cara Mengaktifkan Akses Lintas Akun ke Konsol AWS Manajemen di Blog AWS Keamanan.

[6] Setelah Anda mendapatkan kredensial sementara Anda, Anda tidak dapat mengakses Konsol Manajemen Konsol Manajemen AWS dengan meneruskan kredensial ke titik akhir masuk tunggal gabungan. Lihat informasi yang lebih lengkap di Aktifkan akses broker identitas khusus ke AWS konsol.

Token pembawa layanan

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Layanan ini mendukung protokol yang mengharuskan Anda untuk menggunakan token pembawa alih-alih menggunakan tradisionalAWS Tanda Tangan Versi 4 untuk permintaan API. Saat Anda melakukan AWS CLI atau operasi AWS API yang memerlukan token pembawa, AWS layanan meminta token pembawa atas nama Anda. Layanan memberi Anda token, yang kemudian dapat Anda gunakan untuk melakukan operasi selanjutnya dalam layanan tersebut.

AWS STS token pembawa layanan mencakup informasi dari otentikasi utama asli Anda yang dapat memengaruhi izin Anda. Informasi ini dapat mencakup tanda utama, tanda sesi, dan kebijakan sesi. Access key ID token dimulai dengan awalan `ABIA`. Ini membantu Anda untuk mengidentifikasi operasi yang dilakukan menggunakan token pembawa layanan dalam log CloudTrail anda.

> **Important**
>
> Token pembawa dapat digunakan hanya untuk panggilan ke layanan yang membuatnya dan di Wilayah tempat pembuatannya. Anda tidak dapat menggunakan token pembawa untuk melakukan operasi di layanan atau Wilayah lain.

Contoh layanan yang mendukung token pembawa adalah AWS CodeArtifact. Sebelum Anda dapat berinteraksi dengan AWS CodeArtifact menggunakan manajer paket seperti NPM, Maven, atau PIP, Anda harus memanggil operasi. `aws codeartifact get-authorization-token` Operasi ini mengembalikan token pembawa yang dapat Anda gunakan untuk melakukan AWS CodeArtifact operasi. Atau, Anda dapat menggunakan perintah `aws codeartifact login` yang menyelesaikan operasi yang sama dan kemudian mengonfigurasi klien Anda secara otomatis.

Jika Anda melakukan tindakan dalam AWS layanan yang menghasilkan token pembawa untuk Anda, Anda harus memiliki izin berikut dalam kebijakan IAM Anda:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Untuk contoh token pembawa layanan, lihat Menggunakan kebijakan berbasis identitas untuk AWS CodeArtifact dalam panduan pengguna. AWS CodeArtifact

Minta kredentif keamanan sementara

Untuk meminta kredentif keamanan sementara, Anda dapat menggunakan operasi AWS Security Token Service (AWS STS) di AWS API. Ini termasuk operasi untuk membuat dan menyediakan

kredentif keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke AWS sumber daya Anda. Untuk informasi lebih lanjut tentang AWS STS, lihatKredensial keamanan sementara di IAM. Untuk mempelajari lebih lanjut tentang berbagai metode berbeda yang dapat Anda gunakan untuk meminta kredensial keamanan sementara dengan mengasumsikan peran, lihat Metode untuk mengambil peran.

Untuk memanggil operasi API, Anda dapat menggunakan salah satu SDK AWS. SDK tersedia untuk berbagai bahasa dan lingkungan pemrograman, termasuk Java, .NET, Python, Ruby, Android, dan iOS. SDK menangani tugas seperti menandatangani permintaan Anda secara kriptografis, mencoba kembali permintaan jika perlu, dan menangani respons kesalahan. Anda juga dapat menggunakan AWS STS Query API, yang dijelaskan dalam Referensi AWS Security Token Service API. Akhirnya, dua alat baris perintah mendukung AWS STS perintah: the AWS Command Line Interface, dan the AWS Tools for Windows PowerShell.

Operasi AWS STS API membuat sesi baru dengan kredenSIAL keamanan sementara yang menyertakan pasangan kunci akses dan token sesi. Pasangan pasangan access key terdiri dari access key ID dan secret access key. Pengguna (atau aplikasi yang dijalankan pengguna) dapat menggunakan kredensial ini untuk mengakses sumber daya Anda.

Anda dapat membuat sesi peran dan meneruskan kebijakan sesi dan tag sesi secara terprogram menggunakan operasi AWS STS API. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas peran dan kebijakan sesi. Untuk informasi lebih lanjut tentang kebijakan sesi, lihat Kebijakan sesi. Untuk informasi lebih lanjut tentang tanda sesi, lihat Lewati tag sesi di AWS STS.

> **Note**
>
> Ukuran token sesi yang dikem AWS STS balikan operasi API tidak tetap. Kami sangat menyarankan agar Anda tidak membuat asumsi tentang ukuran maksimum. Ukuran token umumnya kurang dari 4096 byte, tetapi itu dapat bervariasi.

Penggunaan AWS STS dengan AWS Wilayah

Anda dapat mengirim panggilan AWS STS API ke titik akhir global atau ke salah satu titik akhir Regional. Jika Anda memilih titik akhir yang lebih dekat dengan Anda, Anda dapat mengurangi latensi dan meningkatkan kinerja panggilan API Anda. Anda juga dapat memilih untuk mengarahkan panggilan Anda ke titik akhir Regional alternatif jika Anda tidak lagi dapat berkomunikasi dengan titik akhir awal.

Jika Anda menggunakan salah satu dari berbagai AWS SDK, gunakan metode SDK tersebut untuk menentukan Wilayah sebelum Anda melakukan panggilan API. Jika Anda membuat permintaan HTTP API secara manual, maka Anda harus mengarahkan permintaan ke titik akhir yang benar sendiri. Untuk informasi lebih lanjut, lihat bagian AWS STS dari Wilayah dan Titik Akhir dan Kelola AWS STS dalam sebuah Wilayah AWS.

Berikut ini adalah operasi API yang dapat Anda gunakan untuk memperoleh kredenSIAL sementara untuk digunakan di AWS lingkungan dan aplikasi Anda.

Meminta kredenSIAL untuk delegasi lintas akun dan federasi melalui broker identitas khusus

Oper `AssumeRole` asi API berguna untuk memungkinkan pengguna IAM yang ada mengakses AWS sumber daya yang belum mereka akses. Misalnya, pengguna mungkin memerlukan akses ke sumber daya di sumber daya lain Akun AWS. Itu juga berguna sebagai sarana untuk mendapatkan akses istimewa sementara waktu—misalnya, untuk menyediakan Multi-Factor Authentication (MFA) Anda harus memanggil API ini menggunakan kredenSIAL aktif. Untuk mempelajari siapa yang dapat memanggil operasi ini, lihatBandingkan AWS STS kredensialnya. Untuk informasi selengkapnya, lihat Buat peran untuk memberikan izin kepada pengguna IAM dan Akses API aman dengan MFA.

Untuk meminta kredenSIAL keamanan sementara untuk delegasi lintas akun dan federasi melalui broker identitas khusus

1. Otentikasi dengan kredentif AWS keamanan Anda. Panggilan ini harus dilakukan dengan menggunakan kredensial keamanan AWS yang berlaku.
2. Panggil operasi `AssumeRole`.

Contoh berikut menunjukkan permintaan sampel dan respons menggunakan `AssumeRole`. Contoh permintaan ini mengasumsikan peran `demo` untuk durasi yang ditentukan dengan disertakan Kebijakan sesi, tanda sesi, ID eksternal, dan identitas sumber. Sesi yang dihasilkan diberi nama `John-session`.

Example Contoh permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
```

```
&RoleArn=arn:aws:iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17       %22%2C%22Statement%22%3A%5B%7B%22Sid
%22%3A%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*
%22%2C%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

Nilai kebijakan yang ditampilkan dalam contoh sebelumnya adalah URL-encoded versi kebijakan berikut:

JSON

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

Parameter AUTHPARAMS dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan permintaan API AWS HTTP. Kami menyarankan menggunakan SDK AWS untuk membuat permintaan API, dan satu keuntungan dalam melakukannya adalah SDK mengurus tanda tangan permintaan untuk Anda. Jika Anda harus membuat dan menandatangani permintaan API secara manual, lihat Menandatangani Per AWS mintaan Dengan Menggunakan Tanda Tangan Versi 4 di Referensi Umum Amazon Web Services untuk mempelajari cara menandatangani permintaan.

Selain kredensial keamanan sementara, responsnya mencakup Amazon Resource Name (ARN) untuk pengguna federasi dan waktu kedaluwarsa kredensial tersebut.

Example Contoh tanggapan

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
```

```
    AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
    LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
    QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
    9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
    +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
    wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>34</PackedPolicySize>
<SessionTokenSize>1428</SessionTokenSize>
<SessionTokenUtilization>34</SessionTokenUtilization>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Kebijakan sesi dan tag sesi menambah ukuran token sesi. Jika token melebihi batas ukuran, permintaan Anda gagal, bahkan jika plaintext Anda berada dalam batas lain. Elem `SessionTokenUtilization` en respons menunjukkan berapa persentase dari batas ukuran yang digunakan token. Elem `PackedPolicySize` en respons tidak digunakan lagi. Gunakan `SessionTokenUtilization` sebagai gantinya.

Meminta kredenSIAL melalui penyedia OIDC

Operasi `AssumeRoleWithWebIdentity` API mengembalikan satu set kredenSIAL AWS keamanan sementara dengan imbalan JSON Web Token (JWT). Ini termasuk penyedia identitas publik, seperti Login dengan Amazon, Facebook, Google, dan penyedia yang mengeluarkan JWT yang kompatibel dengan penemuan OpenID Connect (OIDC), seperti GitHub tindakan atau Azure Devops. Untuk informasi selengkapnya, lihat Federasi OIDC.

> **Note**
>
> `AssumeRoleWithWebIdentity`permintaan tidak ditandatangani dengan, dan tidak memerlukan AWS kredensi.

Meminta kredenSIAL melalui penyedia OIDC

1. Panggil operasi `AssumeRoleWithWebIdentity`.

   Saat Anda meneleponAssumeRoleWithWebIdentity, AWS memvalidasi token yang disajikan dengan memverifikasi tanda tangan digital menggunakan kunci publik yang tersedia melalui keyset web JSON (JWKS) IdP Anda. Jika token valid, dan semua kondisi yang ditetapkan dalam kebijakan kepercayaan peran IAM terpenuhi, AWS mengembalikan informasi berikut kepada Anda:

   - Serangkaian kredensial keamanan sementara. Ini terdiri dari access key ID, secret access key, dan token sesi.
   - ID peran dan ARN peran yang diasumsikan.
   - Suatu nilai `SubjectFromWebIdentityToken` yang memuat ID pengguna unik.
2. Aplikasi Anda kemudian dapat menggunakan kredentif keamanan sementara yang dikembalikan sebagai respons untuk melakukan panggilan AWS API. Ini adalah proses yang sama seperti membuat panggilan AWS API dengan kredenSIAL keamanan jangka panjang. Perbedaannya adalah Anda harus menyertakan token sesi, yang memungkinkan AWS memverifikasi bahwa kredenSIAL keamanan sementara valid.

Aplikasi Anda harus menyimpan kredenSIAL yang dikembalikan oleh AWS STS dan menyegarkannya sesuai kebutuhan. Jika aplikasi Anda dibuat menggunakan AWS SDK, SDK memiliki penyedia kredensia yang dapat menangani kredensi panggilan `AssumeRoleWithWebIdentity` dan penyegaran sebelum ked AWS aluwarsa. Untuk informasi selengkapnya, lihat AWS SDK dan penyedia kredensia standar Alat di Panduan Refer AWS ensi SDK dan Alat.

Meminta kredenSIAL melalui penyedia identitas SAML 2.0

Operasi `AssumeRoleWithSAML` API mengembalikan satu set kredentif keamanan sementara untuk prinsipal federasi SAML yang diautentikasi oleh sistem identitas organisasi Anda yang ada.

Pengguna juga harus menggunakan SAML 2.0 (Security Assertion Markup Language) untuk menyampaikan informasi autentikasi dan otorisasi ke AWS. Operasi API ini berguna pada organisasi yang telah mengintegrasikan sistem identitas mereka (seperti Windows Active Directory atau OpenLDAP) dengan perangkat lunak yang dapat menghasilkan pernyataan SAML. Integrasi tersebut menyediakan informasi tentang identitas dan izin pengguna (seperti ctive Directory Federation Services atau Shibboleth). Untuk informasi selengkapnya, lihat Federasi SAML 2.0.

1. Panggil operasi `AssumeRoleWithSAML`.

   Ini adalah panggilan yang tidak ditandatangani, artinya Anda tidak perlu mengotentikasi kredentif AWS keamanan sebelum membuat permintaan.

   > **Note**
   >
   > Sebuah panggilan ke `AssumeRoleWithSAML` tidak ditandatangani (dienkripsi). Oleh karena itu, Anda hanya boleh menyertakan kebijakan sesi opsional jika permintaan dikirim melalui perantara terpercaya. Dalam hal ini, seseorang dapat mengubah kebijakan untuk menghapus pembatasan.

2. Saat Anda menelep `AssumeRoleWithSAML` on, AWS memverifikasi keaslian pernyataan SAML. Dengan asumsi bahwa penyedia identitas memvalidasi pernyataan, AWS mengembalikan informasi berikut kepada Anda:

   - Serangkaian kredensial keamanan sementara. Ini terdiri dari access key ID, secret access key, dan token sesi.
   - ID peran dan ARN peran yang diasumsikan.
   - Nilai `Audience` yang berisi nilai atribut `Recipient` dari `SubjectConfirmationData` elemen pernyataan SAML.
   - Nilai `Issuer` yang berisi nilai dari elemen `Issuer` pernyataan SAML.
   - `NameQualifier`Elemen yang berisi nilai hash yang dibangun dari `Issuer` nilai, Akun AWS ID, dan nama ramah penyedia SAML. Ketika dikombinasikan dengan `Subject` elemen, mereka dapat secara unik mengidentifikasi prinsipal federasi SAML.
   - Suatu Elemen `Subject` yang berisi nilai dari elemen `NameID` dalam elemen `Subject` pernyataan SAML.
   - Suatu elemen `SubjectType` yang menunjukkan format elemen `Subject`. Kunci ini dapat berupa `persistent`, `transient`, atau URI `Format` penuh dari elemen `Subject` dan

`NameID` yang digunakan dalam pernyataan SAML Anda. Untuk informasi tentang elemen `NameID Format` atribut, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

3. Gunakan kredentif keamanan sementara yang dikembalikan dalam respons untuk melakukan panggilan AWS API. Ini adalah proses yang sama seperti membuat panggilan AWS API dengan kredenSIAL keamanan jangka panjang. Perbedaannya adalah bahwa Anda harus menyertakan token sesi, yang memungkinkan AWS memverifikasi bahwa kredensial keamanan sementara valid.

Aplikasi Anda harus menyimpan kredensial. Secara default kredensial kedaluwarsa setelah satu jam. Jika Anda tidak menggunakan AmazonSTSCredentialsProvider tindakan di AWS SDK, terserah Anda dan aplikasi Anda untuk menelepon `AssumeRoleWithSAML` lagi. Panggil operasi ini untuk mendapatkan rangkaian kredensial keamanan sementara yang baru sebelum kredensial lama kedaluwarsa.

Meminta kredenSIAL melalui broker identitas khusus

Operasi `GetFederationToken` API mengembalikan satu set kredenSIAL keamanan sementara untuk prinsi AWS STS pal pengguna federasi. API ini berbeda dari `AssumeRole` karena periode kedaluwarsa default jauh lebih lama (12 jam, bukan satu jam). Sebagai tambahan, Anda dapat menggunakan parameter `DurationSeconds` untuk menentukan durasi untuk kredensial keamanan sementara agar tetap valid. KredenSIAL yang dihasilkan berlaku untuk durasi yang ditentukan, antara 900 detik (15 menit) hingga 129.600 detik (36 jam). Periode kedaluwarsa yang lebih lama dapat membantu mengurangi jumlah panggilan ke AWS karena Anda tidak perlu mendapatkan kredenSIAL baru sesering mungkin.

1. Otentikasi dengan kredentif AWS keamanan pengguna IAM spesifik Anda. Panggilan ini harus dilakukan menggunakan kredentif AWS keamanan yang valid.
2. Panggil operasinya `GetFederationToken`.

Pang `GetFederationToken` gilan mengembalikan kredentif keamanan sementara yang terdiri dari token sesi, kunci akses, kunci rahasia, dan kedaluwarsa. Anda dapat menggunakan `GetFederationToken` jika anda ingin mengelola izin di dalam organization anda (misalnya, menggunakan aplikasi proksi untuk menetapkan izin).

Contoh berikut menunjukkan permintaan sampel dan respons yang menggunakan `GetFederationToken`. Permintaan contoh ini meminta menggabungkan pengguna panggilan untuk

durasi yang ditentukan dengan ARN kebijakan sesi dantanda sesi. Sesi yang dihasilkan diberi nama `Jane-session`.

Example Contoh permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

Kebijakan ARN yang ditunjukkan pada contoh sebelumnya mencakup AR URL-encoded N berikut:

`arn:aws:iam::123456789012:policy/Role1policy`

Selain itu, perhatikan bahwa parameter &AUTHPARAMS dalam contoh dimaksudkan sebagai kerangka untuk informasi autentikasi. Ini adalah tanda tangan, yang harus Anda sertakan dengan permintaan API AWS HTTP. Kami menyarankan menggunakan SDK AWS untuk membuat permintaan API, dan satu keuntungan dalam melakukannya adalah SDK mengurus tanda tangan permintaan untuk Anda. Jika Anda harus membuat dan menandatangani permintaan API secara manual, lihat Menandatangani Per AWS mintaan Dengan Menggunakan Tanda Tangan Versi 4 di Referensi Umum Amazon Web Services untuk mempelajari cara menandatangani permintaan.

Selain kredensial keamanan sementara, responsnya mencakup Amazon Resource Name (ARN) untuk pengguna federasi dan waktu kedaluwarsa kredensial tersebut.

Example Contoh tanggapan

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
```

```
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>26</PackedPolicySize>
<SessionTokenSize>1105</SessionTokenSize>
<SessionTokenUtilization>26</SessionTokenUtilization>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Kebijakan sesi dan tag sesi menambah ukuran token sesi. Jika token melebihi batas ukuran, permintaan Anda gagal, bahkan jika plaintext Anda berada dalam batas lain. Elem `SessionTokenUtilization` en respons menunjukkan berapa persentase dari batas ukuran yang digunakan token. Elem `PackedPolicySize` en respons tidak digunakan lagi. Gunakan `SessionTokenUtilization` sebagai gantinya.

AWS merekomendasikan agar Anda memberikan izin di tingkat sumber daya (misalnya, Anda melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3), Anda dapat menghilangkan parameter tersebut. `Policy` Namun, jika Anda tidak menyertakan kebijakan untuk prinsipal pengguna AWS STS federasi, kredentif keamanan sementara tidak akan memberikan izin apa pun. Dalam kasus ini, Anda harus menggunakan kebijakan sumber daya untuk memberikan akses pengguna gabungan ke sumber daya AWS Anda.

Misalnya, asumsikan Akun AWS nomor Anda adalah 111122223333, dan Anda memiliki bucket Amazon S3 yang ingin Anda izinkan untuk mengakses Susan. Kredensial keamanan sementara Susan tidak menyertakan kebijakan untuk bucket. Dalam hal ini, Anda perlu

memastikan bahwa bucket memiliki kebijakan ARN yang cocok dengan ARN Susan, seperti `arn:aws:sts::111122223333:federated-user/Susan`.

Meminta kredensi untuk pengguna di lingkungan yang tidak tepercaya

Operasi API `GetSessionToken` mengembalikan satu set kredensial keamanan sementara untuk pengguna IAM yang sudah ada. Ini berguna untuk memberikan keamanan yang ditingkatkan, seperti mengizinkan AWS permintaan hanya ketika MFA diaktifkan untuk pengguna IAM. Karena kredensial bersifat sementara, kredensial ini memberikan keamanan yang lebih baik ketika Anda memiliki pengguna IAM yang mengakses sumber daya Anda melalui lingkungan yang kurang aman. Contoh lingkungan yang kurang aman termasuk perangkat seluler atau browser web.

1. Otentikasi dengan kredentif AWS keamanan pengguna IAM spesifik Anda. Panggilan ini harus dilakukan menggunakan kredentif AWS keamanan yang valid.
2. Panggil operasinya `GetSessionToken`.
3. `GetSessionToken`mengembalikan kredentif keamanan sementara yang terdiri dari token sesi, ID kunci akses, dan kunci akses rahasia.

Secara default, kredensial keamanan sementara untuk pengguna IAM berlaku selama maksimal 12 jam. Tetapi Anda dapat meminta durasi sesingkat 15 menit atau selama 36 jam menggunakan parameter `DurationSeconds`. Untuk alasan keamanan, token untuk an Pengguna root akun AWS dibatasi untuk durasi satu jam.

Contoh berikut menunjukkan permintaan sampel dan respons menggunakan `GetSessionToken`. Jawaban juga mencakup waktu kedaluwarsa kredensial keamanan sementara.

Example Contoh Permintaan

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

Parameter `AUTHPARAMS` dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan permintaan API AWS HTTP. Kami menyarankan menggunakan SDK AWS untuk membuat permintaan API, dan satu keuntungan dalam melakukannya adalah SDK mengurus tanda tangan permintaan untuk Anda. Jika Anda harus

membuat dan menandatangani permintaan API secara manual, lihat Menandatangani Per AWS mintaan Dengan Menggunakan Tanda Tangan Versi 4 di Referensi Umum Amazon Web Services untuk mempelajari cara menandatangani permintaan.

Example Contoh tanggapan

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<SessionTokenSize>856</SessionTokenSize>
<SessionTokenUtilization>20</SessionTokenUtilization>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

Secara opsional, `GetSessionToken` permintaan dapat menyertakan `SerialNumber` dan `TokenCode` nilai untuk verifikasi otentikasi AWS multi-faktor (MFA). Jika nilai yang diberikan valid, AWS STS berikan kredentif keamanan sementara yang menyertakan status otentikasi MFA. KredenSIAL keamanan sementara kemudian dapat digunakan untuk mengakses operasi MFA-protected API atau AWS situs web selama otentikasi MFA valid.

Contoh berikut menunjukkan permintaan `GetSessionToken` yang menyertakan kode verifikasi MFA dan nomor seri perangkat.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
```

```
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> Panggilan ke AWS STS bisa ke titik akhir global atau ke salah satu titik akhir Regional yang Anda Akun AWS aktifkan. Untuk informasi lebih lanjut, lihat bagian AWS STS pada Wilayah dan Titik Akhir.
>
> Parameter AUTHPARAMS dalam contoh adalah placeholder untuk tanda tangan Anda. Tanda tangan adalah informasi otentikasi yang harus Anda sertakan dengan permintaan API AWS HTTP. Kami menyarankan menggunakan SDK AWS untuk membuat permintaan API, dan satu keuntungan dalam melakukannya adalah SDK mengurus tanda tangan permintaan untuk Anda. Jika Anda harus membuat dan menandatangani permintaan API secara manual, lihat Menandatangani Per AWS mintaan Dengan Menggunakan Tanda Tangan Versi 4 di Referensi Umum Amazon Web Services untuk mempelajari cara menandatangani permintaan.

Gunakan kredenSIAL sementara dengan AWS sumber daya

Anda dapat menggunakan kredentif keamanan sementara untuk membuat permintaan terprogram untuk AWS sumber daya menggunakan AWS API AWS CLI or (menggunakan AWS SDK). KredenSIAL sementara memberikan izin yang sama dengan kredenSIAL keamanan jangka panjang, seperti kredenSIAL pengguna IAM. Namun, ada beberapa perbedaan:

- Saat Anda melakukan panggilan menggunakan kredenSIAL keamanan sementara, panggilan harus menyertakan token sesi, yang dikembalikan bersama dengan kredenSIAL sementara tersebut. AWS menggunakan token sesi untuk memvalidasi kredentif keamanan sementara.
- KredenSIAL sementara kedaluwarsa setelah interval tertentu. Setelah kredenSIAL sementara kedaluwarsa, panggilan apa pun yang Anda lakukan dengan kredenSIAL tersebut akan gagal, jadi Anda harus membuat kumpulan kredenSIAL sementara baru. KredenSIAL sementara tidak dapat diperpanjang atau disegarkan di luar interval asli yang ditentukan.
- Saat Anda menggunakan kredensial sementara untuk membuat permintaan, prinsipal Anda mungkin mencakup satu set tanda. Tanda ini berasal dari tanda sesi dan tanda yang dilampirkan pada peran yang Anda asumsikan. Untuk informasi lebih lanjut tentang tanda sesi, lihat Lewati tag sesi di AWS STS.

Jika Anda menggunakan SDK, AWS Command Line Interface (AWS CLI), atau Tools untuk Windows PowerShell, cara mendapatkan dan menggunakan kredenSIAL keamanan sementara berbeda dengan kontek AWS snya. Jika Anda menjalankan PowerShell perintah kode AWS CLI, atau Alat untuk Windows di dalam instans EC2, Anda dapat memanfaatkan peran untuk Amazon EC2. Jika tidak, Anda dapat menghubungi API AWS STS untuk mendapatkan kredensial sementara, kemudian menggunakannya secara eksplisit untuk melakukan panggilan ke layanan AWS .

> **Note**
>
> Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan memberikan kredentif keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke AWS sumber daya Anda. Untuk informasi lebih lanjut tentang AWS STS, lihatKredensial keamanan sementara di IAM. AWS STS adalah layanan global yang memiliki titik akhir default di`https://sts.amazonaws.com`. Titik akhir ini berada di Wilayah AS Timur (Virginia Utara), meskipun kredenSIAL yang Anda dapatkan dari ini dan titik akhir lainnya valid secara global. Kredensial ini bekerja dengan layanan dan sumber daya di Wilayah mana pun. Anda juga dapat memilih untuk melakukan panggilan AWS STS API ke titik akhir di salah satu Wilayah yang didukung. Hal ini dapat mengurangi latensi dengan membuat permintaan dari server di Wilayah yang secara geografis lebih dekat dengan Anda. Tidak peduli dari Wilayah mana kredensial Anda berasal, mereka bekerja secara global. Untuk informasi selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Daftar Isi

- Menggunakan kredensial sementara di instans Amazon EC2
- Menggunakan kredenSIAL keamanan sementara dengan AWS SDK
- Menggunakan kredenSIAL keamanan sementara dengan AWS CLI
- Menggunakan kredensial keamanan sementara dengan operasi API
- Informasi selengkapnya

Menggunakan kredensial sementara di instans Amazon EC2

Jika Anda ingin menjalankan AWS CLI perintah atau kode di dalam instance EC2, cara yang disarankan untuk mendapatkan kredenSIAL adalah dengan menggunakan peran untuk Amazon EC 2. Anda membuat peran IAM yang menentukan izin yang ingin Anda berikan ke aplikasi yang berjalan di instans EC2 Saat Anda meluncurkan instans, Anda mengaitkan peran dengan instans tersebut.

Perintah AWS CLI Aplikasi, dan Alat untuk Windows PowerShell yang berjalan pada instance kemudian dapat memperoleh kredenSIAL keamanan sementara otomatis dari metadata instance. Anda tidak perlu secara eksplisit mendapatkan kredensial kerahasiaan keamanan sementara. AWS SDK, AWS CLI, dan Tools untuk Windows PowerShell secara otomatis mendapatkan kredenSIAL dari EC2 Instance Metadata Service (IMDS) dan menggunakannya. Kredensial sementara memiliki izin yang Anda tentukan untuk peran yang terkait dengan instans.

Untuk informasi selengkapnya dan untuk contoh, lihat berikut ini:

- Menggunakan Peran IAM untuk Memberikan Akses ke Sumber AWS Daya di Amazon Elastic Compute Cloud — AWS SDK untuk Java
- Memberikan Akses Menggunakan Peran IAM — AWS SDK untuk .NET
- Menciptakan Peran — AWS SDK untuk Ruby

Menggunakan kredenSIAL keamanan sementara dengan AWS SDK

Untuk menggunakan kredenSIAL keamanan sementara dalam kode, Anda secara terprogram memanggil AWS STS API seperti `AssumeRole` dan mengekstrak kredenSIAL dan token sesi yang dihasilkan. Anda kemudian menggunakan nilai-nilai tersebut sebagai kredenSIAL untuk panggilan berikutnya ke AWS. Contoh berikut menunjukkan pseudocode untuk cara menggunakan kredenSIAL keamanan sementara jika Anda menggunakan AWS SDK:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Untuk contoh yang ditulis dalam Python (menggunakan AWS SDK for Python (Boto)), lihatBeralih ke peran IAM (AWS API). Contoh ini menunjukkan cara melakukan panggilan `AssumeRole` untuk mendapatkan kredensial keamanan sementara dan kemudian menggunakan kredensial tersebut untuk melakukan panggilan ke Amazon S3

Untuk detail tentang cara memanggil`AssumeRole`,`GetFederationToken`, dan operasi API lainnya, lihat Refer AWS Security Token Service ensi API. Untuk informasi tentang mendapatkan kredensial keamanan sementara dan token sesi dari hasilnya, lihat dokumentasi untuk SDK yang sedang

Anda kerjakan. Anda dapat menemukan dokumentasi untuk semua AWS SDK di halaman AWS dokumentasi utama, di bagian SDK dan Tool kit.

Anda harus memastikan bahwa Anda mendapatkan set kredensial baru sebelum kredensial yang lama kedaluwarsa. Di beberapa SDK, Anda dapat menggunakan penyedia layanan yang mengelola proses penyegaran kredensial untuk Anda; periksa dokumentasi untuk SDK yang Anda gunakan.

Menggunakan kredenSIAL keamanan sementara dengan AWS CLI

Anda dapat menggunakan kredensial keamanan sementara dengan AWS CLI Ini dapat berguna untuk menguji kebijakan.

Dengan menggunakan AWS CLI, Anda dapat memanggil AWS STS API seperti `AssumeRole` atau `GetFederationToken` dan kemudian menangkap output yang dihasilkan. Contoh berikut ini menunjukkan panggilan ke `AssumeRole` yang mengirimkan output ke file. Dalam contoh, `profile` parameter diasumsikan sebagai profil dalam file AWS CLI konfigurasi. Itu juga diasumsikan sebagai kredensial referensi untuk pengguna IAM yang memiliki izin untuk mengambil peran tersebut.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Setelah perintah selesai, Anda dapat mengekstrak ID kunci akses, kunci akses rahasia, dan token sesi dari mana pun Anda merutekannya. Anda dapat melakukannya secara manual atau dengan menggunakan skrip. Kemudian Anda dapat menetapkan nilai-nilai ini ke variabel lingkungan.

Ketika Anda menjalankan AWS CLI perintah, mencari kre AWS CLI denSIAL dalam urutan tertentu —pertama di variabel lingkungan dan kemudian di file konfigurasi. Oleh karena itu, setelah Anda memasukkan kredenSIAL sementara ke dalam variabel lingkungan, kredenSIAL tersebut digunakan secara default. AWS CLI (Jika Anda menentukan `profile` parameter dalam perintah, akan AWS CLI melewatkan variabel lingkungan. Sebagai gantinya, AWS CLI tampilan di file konfigurasi, yang memungkinkan Anda mengganti kredenSIAL dalam variabel lingkungan jika perlu.)

Contoh berikut menunjukkan bagaimana Anda dapat mengatur variabel lingkungan untuk kredenSIAL keamanan sementara dan kemudian memanggil AWS CLI perintah. Karena tidak ada `profile` parameter yang disertakan dalam AWS CLI perintah, mencari kredenSIAL terlebih dahulu di variabel lingkungan dan oleh karena itu menggunakan kredenSIAL sementara. AWS CLI

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
```

```
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Menggunakan kredensial keamanan sementara dengan operasi API

Jika Anda membuat permintaan API HTTPS langsung ke AWS, Anda dapat menandatangani permintaan tersebut dengan kredentif keamanan sementara yang Anda dapatkan dari AWS Security Token Service (AWS STS). Untuk melakukan ini, Anda menggunakan ID kunci akses dan kunci akses rahasia yang Anda terima AWS STS. Anda menggunakan access key ID dan secret access key dengan cara yang sama seperti Anda menggunakan kredensial jangka panjang untuk menandatangani permintaan. Anda juga menambahkan token sesi yang Anda terima ke permintaan API Anda AWS STS. Anda menambahkan token sesi untuk header HTTP atau parameter string kueri bernama `X-Amz-Security-Token`. Anda menambahkan token sesi untuk HTTP header atauparameter string kueri, tetapi tidak keduanya. Untuk informasi selengkapnya tentang penandatanganan permintaan API HTTPS, lihat AWS Men andatangani Permintaan API di Referensi Umum AWS.

Informasi selengkapnya

Untuk informasi selengkapnya tentang penggunaan AWS STS dengan AWS layanan lain, lihat tautan berikut:

- Amazon S3. Lihat Membuat permintaan menggunakan kredenSIAL sementara pengguna IAM atau Membuat permintaan menggunakan kredenSIAL sementara pengguna federasi di Panduan Pengguna Layanan Peny impanan Sederhana Amazon.
- Amazon SNS. Lihat Menggunakan kebijakan berbasis identitas dengan Amazon SNS di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon.
- Amazon SQS. Lihat Identitas dan manajemen akses di Amazon SQS di Panduan Pengembang Layanan Antrian Sederhana Amazon.
- Amazon SimpleDB. Lihat Menggunakan KredenSIAL Keamanan Sementara di Panduan Pengembang Amazon SimpleDB.

Izin untuk kredensial keamanan sementara

Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat dan menyediakan kredenal keamanan sementara kepada pengguna tepercaya yang dapat mengontrol akses ke sumber daya Anda AWS . Untuk informasi lebih lanjut tentang AWS STS, lihatKredensial keamanan sementara di IAM. Setelah AWS STS mengeluarkan kredensial keamanan sementara, kredensial tersebut berlaku selama periode kedaluwarsa dan tidak dapat dicabut. Namun, izin yang diberikan untuk kredensial keamanan sementara dievaluasi setiap kali sebuah permintaan dibuat ketika menggunakan kredensial tersebut, sehingga Anda dapat memperoleh dampak dari mencabut kredensial dengan mengubah hak akses mereka setelah diterbitkan.

Topik berikut menganggap Anda memiliki pengetahuan tentang AWS izin dan kebijakan. Untuk informasi lebih lanjut pada topik ini, lihat Manajemen akses untuk AWS sumber daya.

Topik

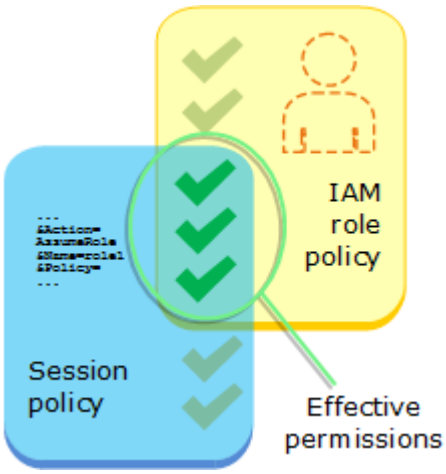
- Izin untuk AssumeRole, AssumeRoleWithSAML, dan AssumeRoleWithWebIdentity
- Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan
- Izin untuk GetFederationToken
- Izin untuk GetSessionToken
- Menonaktifkan izin untuk kredensial keamanan sementara
- Memberikan izin untuk membuat kredensial keamanan sementara
- Memberikan izin untuk menggunakan sesi konsol yang disempurnakan identitas

Izin untuk AssumeRole, AssumeRoleWithSAML, dan AssumeRoleWithWebIdentity

Kebijakan izin dari peran yang sedang diasumsikan menentukan izin untuk kredensial keamanan sementara yang dikembalikan oleh `AssumeRole`, `AssumeRoleWithSAML`, dan `AssumeRoleWithWebIdentity`. Anda menentukan izin ini saat membuat atau memperbarui peran.

Atau, Anda dapat meneruskan kebijakan sesi inline atau terkelola sebagai parameter Operasi API `AssumeRole`, `AssumeRoleWithSAML`, atau `AssumeRoleWithWebIdentity`. Kebijakan sesi membatasi izin untuk sesi kredensial sementara peran tersebut. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas peran dan kebijakan sesi. Anda dapat menggunakan kredensi sementara peran dalam panggilan AWS API berikutnya untuk mengakses sumber daya di akun yang memiliki peran tersebut. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang

diasumsikan. Untuk mempelajari lebih lanjut tentang bagaimana AWS menentukan izin yang efektif dari suatu peran, lihat Logika evaluasi kebijakan.



Kebijakan yang dilampirkan pada kredensi yang membuat panggilan asli tidak `AssumeRole` dievaluasi oleh AWS saat membuat keputusan otorisasi "izinkan" atau "tolak". Pengguna untuk sementara menyerahkan izin asli yang mendukung izin yang ditetapkan oleh peran yang diasumsikan. Dalam kasus operasi `AssumeRoleWithSAML` dan `AssumeRoleWithWebIdentity` API, tidak ada kebijakan untuk dievaluasi karena pemanggil API bukan AWS identitas.

Contoh: Menetapkan izin menggunakan AssumeRole

Anda dapat menggunakan Operasi API `AssumeRole` dengan berbagai jenis kebijakan. Berikut adalah beberapa contoh.

Kebijakan izin peran

Dalam contoh ini, Anda memanggil operasi API `AssumeRole` tanpa menetapkan kebijakan sesi pada opsional parameter `Policy`. Izin yang diberikan untuk kredensial sementara ditentukan oleh kebijakan izin dari peran yang diasumsikan. Contoh kebijakan izin berikut memberikan izin peran untuk mencantumkan semua objek yang terkandung dalam bucket S3 dengan nama `productionapp`. Hal ini juga memungkinkan peran untuk mendapatkan, menempatkan, dan menghapus objek dalam bucket itu.

Example Contoh kebijakan izin peran

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Kebijakan sesi diberikan sebagai parameter

Bayangkan Anda ingin mengizinkan pengguna untuk mengambil peran yang sama seperti dalam contoh sebelumnya. Tetapi dalam hal ini Anda ingin sesi peran hanya memiliki izin untuk mendapatkan dan memasukkan objek ke dalam bucket S3 `productionapp`. Anda tidak ingin mengizinkan mereka untuk menghapus objek. Salah satu cara untuk mencapainya adalah dengan membuat peran baru dan menentukan izin yang diinginkan dalam kebijakan izin peran tersebut. Cara lain untuk mencapai ini adalah dengan memanggil API `AssumeRole` dan menyertakan kebijakan sesi dalam opsional parameter `Policy` sebagai bagian dari operasi API. Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan berbasis identitas peran dan kebijakan sesi. Kebijakan sesi tidak dapat digunakan untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari peran yang sedang diasumsikan. Untuk informasi lebih lanjut tentang izin sesi peran, lihat Kebijakan sesi.

Setelah Anda menerima kredensial sementara sesi baru, Anda dapat memberikannya kepada pengguna yang Anda ingin untuk memiliki izin tersebut.

Misalnya, bayangkan kebijakan berikut ini diberikan sebagai parameter panggilan API. Orang yang menggunakan sesi memiliki izin untuk hanya melakukan tindakan berikut:

- Mencantumkan semua objek ke dalam bucket `productionapp`.
- Mendapatkan dan memasukkan objek ke dalam bucket `productionapp`.

Dalam kebijakan sesi berikut, izin `s3:DeleteObject` difilter dan sesi yang diasumsikan tidak diberikan izin `s3:DeleteObject`. Kebijakan menetapkan izin maksimum untuk sesi peran sehingga menggantikan kebijakan izin yang ada pada peran tersebut.

Example Contoh kebijakan sesi melewati Panggilan API **`AssumeRole`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Resource-based kebijakan

Beberapa AWS sumber daya mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk menentukan izin yang memengaruhi kredensi keamanan sementara. Hanya beberapa sumber daya, seperti bucket Amazon S3, topik Amazon SNS, dan antrean Amazon SQS mendukung kebijakan berbasis sumber daya. Contoh berikut memperluas contoh sebelumnya, menggunakan bucket S3 bernama `productionapp`. Kebijakan berikut ini terlampir di bucket.

Ketika Anda melampirkan kebijakan berbasis sumber daya berikut ini ke bucket `productionapp`, semua pengguna tidak diberi izin untuk menghapus objek dari bucket. (Lihat elemen `Principal` dalam kebijakan.) Ini termasuk semua pengguna peran yang diasumsikan, meskipun kebijakan

izin peran memberikan izin `DeleteObject`. Sebuah pernyataan `Deny` yang jelas selalu lebih diutamakan daripada pernyataan `Allow`.

Example Contoh kebijakan bucket

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Untuk informasi selengkapnya tentang bagaimana beberapa jenis kebijakan digabungkan dan dievaluasi oleh AWS, lihatLogika evaluasi kebijakan.

Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan

Peran IAM adalah objek di IAM yang diberi izin. Bila Anda mengasumsikan peran tersebut menggunakan identitas IAM atau identitas dari luar AWS, Anda menerima sesi dengan izin yang ditetapkan untuk peran tersebut.

Saat Anda melakukan tindakan AWS, informasi tentang sesi Anda dapat dicatat AWS CloudTrail untuk dipantau oleh administrator akun Anda. Administrator dapat mengkonfigurasi peran untuk meminta identitas untuk melewati string kustom yang mengidentifikasi orang atau aplikasi yang melakukan tindakan di AWS. Informasi identitas ini disimpan sebagai identitas sumber di AWS CloudTrail. Ketika administrator meninjau aktivitas di CloudTrail, mereka dapat melihat informasi identitas sumber untuk menentukan siapa atau tindakan apa yang melakukan dengan sesi peran yang diasumsikan.

Setelah identitas sumber ditetapkan, itu hadir dalam permintaan untuk AWS tindakan apa pun yang diambil selama sesi peran. Nilai yang ditetapkan tetap ada ketika peran digunakan untuk mengambil peran lain melalui AWS API AWS CLI or, yang dikenal sebagai rantai peran. Nilai yang ditetapkan tidak dapat diubah selama sesi peran.

Administrator dapat mengonfigurasi izin granular berdasarkan keberadaan atau nilai identitas sumber untuk mengontrol AWS tindakan lebih lanjut yang diambil dengan peran bersama. Anda dapat memutuskan apakah atribut identitas sumber dapat digunakan, apakah itu diperlukan, dan apakah nilai dapat digunakan.

Cara Anda menggunakan identitas sumber berbeda dari nama sesi peran dan tanda sesi dalam cara yang penting. Nilai identitas sumber tidak dapat diubah setelah ditetapkan, dan tetap ada untuk tindakan tambahan yang diambil dengan sesi peran. Berikut cara menggunakan tanda sesi dan nama sesi peran:

- Tag sesi — Anda dapat meneruskan tag sesi saat Anda mengambil peran atau menyatukan pengguna. Tanda sesi hadir ketika peran diasumsikan. Kemudian, Anda dapat menentukan kebijakan yang menggunakan kunci syarat tanda untuk memberikan izin kepada penanggung jawab Anda berdasarkan tanda mereka. Kemudian Anda dapat menggunakan CloudTrail untuk melihat permintaan yang dibuat untuk mengambil peran atau menggabungkan pengguna. Untuk mempelajari lebih lanjut tentang tanda sesi, lihat Lewati tag sesi di AWS STS.
- Nama sesi peran — Anda dapat menggunakan kunci `sts:RoleSessionName` kondisi dalam kebijakan kepercayaan peran untuk mengharuskan pengguna memberikan nama sesi tertentu saat mereka mengambil peran. Nama sesi peran dapat digunakan untuk membedakan sesi peran ketika peran digunakan oleh penanggung jawab yang berbeda. Untuk mempelajari lebih lanjut tentang nama sesi peran, lihat sts: RoleSessionName.

Kami menyarankan Anda menggunakan identitas sumber ketika Anda ingin mengontrol identitas yang mengasumsikan peran. Identitas sumber juga berguna untuk menambang CloudTrail log untuk menentukan siapa yang menggunakan peran untuk melakukan tindakan.

Topik

- Menyiapkan untuk menggunakan identitas sumber
- Hal yang perlu diketahui tentang identitas sumber
- Izin yang diperlukan untuk menetapkan identitas sumber
- Menentukan identitas sumber ketika mengasumsikan peran
- Menggunakan identitas sumber dengan AssumeRole
- Menggunakan identitas sumber dengan AssumeRoleWithSAML
- Menggunakan identitas sumber dengan AssumeRoleWithWebIdentity
- Mengontrol akses menggunakan informasi identitas sumber

- Melihat identitas sumber di CloudTrail

Menyiapkan untuk menggunakan identitas sumber

Cara Anda mengatur untuk menggunakan identitas sumber tergantung metode yang digunakan ketika peran Anda diasumsikan. Misalnya, pengguna IAM Anda mungkin mengambil peran secara langsung menggunakan `AssumeRole` operasi. Jika Anda memiliki identitas perusahaan, juga dikenal sebagai identitas tenaga kerja, mereka mungkin mengakses AWS sumber daya Anda menggunakan`AssumeRoleWithSAML`. Jika pengguna akhir mengakses aplikasi seluler atau web Anda, mereka mungkin melakukannya menggunakan `AssumeRoleWithWebIdentity`. Berikut ini adalah gambaran umum alur kerja tingkat tinggi untuk membantu Anda memahami bagaimana Anda dapat mengatur untuk memanfaatkan informasi identitas sumber di lingkungan yang ada.

1. Konfigurasikan pengguna dan peran pengujian — Menggunakan lingkungan praproduksi, konfigurasikan pengguna dan peran uji, serta konfigurasikan kebijakan mereka untuk memungkinkan pengaturan identitas sumber.

   Jika Anda menggunakan penyedia identitas (IdP) untuk identitas federasi Anda, konfigurasikan IdP Anda untuk melewati atribut pengguna pilihan Anda untuk identitas sumber dalam pernyataan atau token.
2. Asumsikan peran — Uji asumsi peran dan meneruskan identitas sumber dengan pengguna dan peran yang Anda atur untuk pengujian.
3. Tinjau CloudTrail — Tinjau informasi identitas sumber untuk peran pengujian Anda di CloudTrail log Anda.
4. Latih pengguna Anda — Setelah Anda menguji di lingkungan praproduksi Anda, pastikan bahwa pengguna Anda tahu cara meneruskan informasi identitas sumber, jika perlu. Tetapkan tenggat waktu untuk kapan Anda akan meminta pengguna Anda untuk memberikan identitas sumber di lingkungan produksi Anda.
5. Konfigurasikan kebijakan produksi — Konfigurasikan kebijakan Anda untuk lingkungan produksi Anda, lalu tambahkan kebijakan tersebut ke pengguna dan peran produksi Anda.
6. Memantau aktivitas — Pantau aktivitas peran produksi Anda menggunakan CloudTrail log.

Hal yang perlu diketahui tentang identitas sumber

Ingatlah hal-hal berikut ini saat bekerja dengan identitas sumber.

- Kebijakan kepercayaan untuk semua peran yang terhubung ke penyedia identitas (IdP) harus memiliki izin `sts:SetSourceIdentity`. Untuk peran yang tidak memiliki izin ini dalam kebijakan kepercayaan peran, operasi `AssumeRole*` akan gagal. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan identitas sumber. Lalu tambahkan izin `sts:SetSourceIdentity` hanya untuk peran yang terhubung ke IdP yang terpisah.
- Ketika identitas menetapkan identitas sumber, kunci `sts:SourceIdentity` terdapat dalam permintaan. Untuk tindakan selanjutnya yang diambil selama sesi peran, kunci `aws:SourceIdentity` terdapat dalam permintaan. AWS tidak mengontrol nilai identitas sumber di salah satu kunci `sts:SourceIdentity` atau `aws:SourceIdentity`. Jika Anda memilih untuk meminta identitas sumber, Anda harus memilih atribut yang Anda inginkan pengguna atau IdP untuk menyediakan. Untuk tujuan keamanan, Anda harus memastikan bahwa Anda dapat mengontrol bagaimana nilai-nilai tersebut disediakan.
- Nilai identitas sumber harus antara 2 dan 256 karakter, hanya dapat berisi karakter alfanumerik, garis bawah, dan karakter berikut:., + = @ - (tanda hubung). Anda tidak dapat menggunakan nilai yang dimulai dengan teks **`aws:`**. Awalan ini dicadangkan untuk penggunaan AWS internal.
- Informasi identitas sumber tidak ditangkap CloudTrail ketika peran AWS layanan atau layanan terkait melakukan tindakan atas nama identitas federasi atau tenaga kerja.

> **Important**
>
> Anda tidak dapat beralih ke peran Konsol Manajemen AWS yang memerlukan identitas sumber untuk ditetapkan saat peran diasumsikan. Untuk mengambil peran seperti itu, Anda dapat menggunakan AWS API AWS CLI or untuk memanggil `AssumeRole` operasi dan menentukan parameter identitas sumber.

Izin yang diperlukan untuk menetapkan identitas sumber

Selain tidakan yang sesuai dengan operasi API, Anda harus memiliki tindakan khusus izin berikut dalam kebijakan Anda:

```
sts:SetSourceIdentity
```

- Untuk menentukan identitas sumber, prinsipal (pengguna dan peran IAM) harus memiliki izin untuk. `sts:SetSourceIdentity` Sebagai administrator, Anda dapat mengonfigurasi ini dalam kebijakan kepercayaan peran dan kebijakan izin penanggung jawab.
- Ketika Anda mengambil peran dengan peran lain, disebut perangkaian peran, izin untuk `sts:SetSourceIdentity` diperlukan dalam kebijakan perizinan penanggung jawab yang mengasumsikan peran dan dalam kebijakan kepercayaan peran pada peran target. Jika tidak, peran operasi asumsi akan gagal.
- Saat menggunakan identitas sumber, kebijakan kepercayaan peran untuk semua peran yang terhubung ke IdP harus memiliki izin `sts:SetSourceIdentity`. Operasi `AssumeRole*` akan gagal untuk peran apa pun yang terhubung ke IdP tanpa izin ini. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan identitas sumber dan menambahkan izin `sts:SetSourceIdentity` hanya untuk peran yang terhubung ke IdP yang terpisah.
- Untuk menetapkan identitas sumber di batas akun, Anda harus menyertakan izin `sts:SetSourceIdentity` di dua tempat. Harus berada dalam kebijakan izin penanggung jawab di akun asal dan kebijakan kepercayaan peran dalam akun target. Anda mungkin perlu melakukannya, misalnya, ketika peran digunakan untuk mengambil peran di akun lain dengan perangkaian peran.

Sebagai administrator akun, bayangkan Anda ingin mengizinkan pengguna IAM `DevUser` di akun Anda untuk mengasumsikan `Developer_Role` di akun yang sama. Tetapi Anda ingin mengizinkan tindakan ini hanya jika pengguna telah menetapkan identitas sumber ke nama pengguna IAM mereka. Anda dapat melampirkan kebijakan berikut ke pengguna IAM.

Example Contoh kebijakan berbasis identitas yang dilampirkan DevUser

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
```

```
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

Untuk memberlakukan nilai identitas sumber yang dapat diterima, Anda dapat mengonfigurasi kebijakan kepercayaan peran berikut. Kebijakan ini memberikan `DevUser` izin pengguna IAM untuk mengambil peran dan menetapkan identitas sumber. Kunci syarat `sts:SourceIdentity` mendefinisikan nilai identitas sumber yang dapat diterima.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
```

```
    }
  ]
}
```

Menggunakan kredenSIAL untuk pengguna IAM`DevUser`, pengguna mencoba untuk mengasumsikan `DeveloperRole` penggunaan AWS CLI permintaan berikut.

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Saat AWS mengevaluasi permintaan, konteks permintaan berisi `sts:SourceIdentity` dari`DevUser`.

Menentukan identitas sumber ketika mengasumsikan peran

Anda dapat menentukan identitas sumber saat menggunakan salah satu operasi AWS STS `AssumeRole*` API untuk mendapatkan kredenSIAL keamanan sementara untuk suatu peran. Operasi API yang Anda gunakan berbeda tergantung kasus penggunaan Anda.

Misalnya, jika Anda menggunakan peran IAM untuk memberi pengguna IAM akses ke AWS sumber daya yang biasanya tidak mereka akses, Anda dapat menggunakan operasi tersebut`AssumeRole`. Jika Anda menggunakan federasi identitas perusahaan untuk mengelola pengguna tenaga kerja Anda, Anda dapat menggunakan operasi `AssumeRoleWithSAML`. Jika Anda menggunakan federasi OIDC untuk mengizinkan pengguna akhir mengakses aplikasi seluler atau web Anda, Anda dapat menggunakan `AssumeRoleWithWebIdentity` operasi. Bagian berikut menjelaskan cara menggunakan identitas sumber dengan setiap operasi. Untuk mempelajari selengkapnya tentang skenario umum untuk kredensial sementara, lihat Skenario umum untuk kredensial sementara.

Menggunakan identitas sumber dengan AssumeRole

`AssumeRole`Operasi mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredentif pengguna atau peran IAM untuk memanggil`AssumeRole`. Untuk meneruskan identitas sumber saat mengambil peran, gunakan `--source-identity` AWS CLI opsi atau parameter `SourceIdentity` AWS API. Contoh berikut menunjukkan cara menentukan identitas sumber menggunakan AWS CLI.

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Menggunakan identitas sumber dengan AssumeRoleWithSAML

Prinsipal yang memanggil `AssumeRoleWithSAML` operasi diautentikasi menggunakan SAML-based federasi. Operasi ini mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang menggunakan SAML-based federasi untuk Konsol Manajemen AWS akses, lihatMengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS. Untuk detail tentang AWS CLI atau akses AWS API, lihatFederasi SAML 2.0. Untuk tutorial menyiapkan federasi SAML untuk pengguna Active Directory Anda, lihat Otentikasi AWS Federasi dengan Layanan Federasi Direktori Aktif (ADFS) di Blog Keamanan. AWS

Sebagai administrator, Anda dapat mengizinkan anggota direktori perusahaan Anda untuk bergabung AWS menggunakan AWS STS `AssumeRoleWithSAML` operasi. Untuk melakukannya, Anda harus menyelesaikan tugas berikut:

1. Mengonfigurasi penyedia SAML di organisasi Anda.
2. Buat penyedia SAML di I AM.
3. Konfigurasikan peran dan izinnya AWS untuk prinsipal federasi SAML Anda.
4. Selesaikan konfigurasi SAML IdP dan buat pernyataan untuk respons autentikasi SAML

Untuk mengatur atribut SAML untuk identitas sumber, termasuk elemen `Attribute` dengan atribut `Name` yang ditetapkan ke `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Gunakan elemen `AttributeValue` untuk menentukan nilai identitas sumber. Misalnya, anggap Anda ingin meneruskan atribut identitas berikut sebagai identitas sumber.

`SourceIdentity:DiegoRamirez`

Untuk meneruskan atribut ini, sertakan elemen berikut dalam pernyataan SAML Anda.

Example Contoh snippet pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
```

```
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Menggunakan identitas sumber dengan AssumeRoleWithWebIdentity

`AssumeRoleWithWebIdentity`Operasi utama yang memanggil diautentikasi menggunakan federasi yang sesuai dengan OpenID Connect (OIDC). Operasi ini menghasilkan serangkaian kredensial sementara yang dapat Anda gunakan untuk mengakses sumber daya AWS . Untuk informasi selengkapnya tentang menggunakan federasi OIDC untuk Konsol Manajemen AWS akses, lihatFederasi OIDC.

Untuk meneruskan identitas sumber dari OpenID Connect (OIDC), Anda harus menyertakan identitas sumber dalam JSON Web Token (JWT). Sertakan identitas sumber dalam namespace `https://aws.amazon.com/` `source_identity` dalam token ketika Anda mengajukan permintaan `AssumeRoleWithWebIdentity`. Untuk mempelajari lebih lanjut tentang token dan klaim OIDC, lihat Menggunakan Token dengan Kolam Pengguna di Panduan Developer Amazon Cognito .

Misalnya, JWT yang diterjemahkan berikut adalah token yang digunakan untuk memanggil `AssumeRoleWithWebIdentity` dengan sumber identitas `Admin`.

Example Contoh Token Web JSON yang diterjemahkan

```
{
    "sub": "john",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Mengontrol akses menggunakan informasi identitas sumber

Ketika identitas sumber awalnya ditetapkan, SourceIdentity kunci sts: hadir dalam permintaan. Setelah identitas sumber ditetapkan, SourceIdentity kunci aws: hadir di semua permintaan berikutnya yang dibuat selama sesi peran. Sebagai administrator, Anda dapat menulis kebijakan yang memberikan otorisasi bersyarat untuk melakukan AWS tindakan berdasarkan keberadaan atau nilai atribut identitas sumber.

Bayangkan bahwa Anda ingin meminta pengembang Anda untuk menetapkan identitas sumber untuk mengambil peran penting yang memiliki izin untuk menulis ke AWS sumber daya produksi yang penting. Bayangkan juga bahwa Anda memberikan AWS akses ke identitas tenaga kerja Anda menggunakan`AssumeRoleWithSAML`. Anda hanya ingin developer senior Saanvi dan Diego memiliki akses ke peran, sehingga Anda membuat kebijakan kepercayaan berikut untuk peran tersebut.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber (SAML)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
```

```
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Kebijakan kepercayaan berisi kondisi `sts:SourceIdentity` yang membutuhkan identitas sumber Saanvi atau Diego untuk mengambil peran penting.

Atau, jika Anda menggunakan penyedia OIDC untuk federasi dan pengguna diautentikasi dengan`AssumeRoleWithWebIdentity`, kebijakan kepercayaan peran Anda mungkin terlihat sebagai berikut.

Example Contoh kebijakan kepercayaan peran untuk identitas sumber (penyedia OIDC)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
```

```
          ]
        }
      }
    }
  ]
}
```

Peran perangkauan dan persyaratan lintas akun

Bayangkan Anda ingin mengizinkan pengguna yang mengasumsikan `CriticalRole` untuk mengambil `CriticalRole_2` di akun lain. Kredensial sesi peran yang diperoleh untuk mengasumsikan `CriticalRole` digunakan untuk merangkai peran untuk peran kedua, `CriticalRole_2`, di akun yang berbeda. Peran sedang diasumsikan di batas akun. Oleh karena itu, izin `sts:SetSourceIdentity` harus diberikan dalam kedua kebijakan izin pada `CriticalRole` dan dalam kebijakan kepercayaan peran di `CriticalRole_2`.

Example Contoh kebijakan izin pada CriticalRole

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Untuk mengamankan pengaturan sumber identitas di batas akun, kebijakan kepercayaan peran berikut hanya mempercayai peran utama untuk `CriticalRole` guna mengatur identitas sumber.

Example Contoh kebijakan kepercayaan peran pada CriticalRole _2

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

Pengguna membuat panggilan berikut menggunakan kredentif sesi peran yang diperoleh dari asumsi CriticalRole. Identitas sumber ditetapkan selama asumsi CriticalRole, jadi tidak perlu disetel lagi secara eksplisit. Jika pengguna mencoba untuk mengatur identitas sumber yang berbeda dari nilai yang ditetapkan ketika diasumsikan `CriticalRole`, permintaan peran asumsi akan ditolak.

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Ketika penanggung jawab panggilan mengasumsikan peran, identitas sumber dalam permintaan tetap dari sesi peran diasumsikan pertama. Oleh karena itu, kedua kunci `aws:SourceIdentity` dan `sts:SourceIdentity` terdapat dalam konteks permintaan.

Melihat identitas sumber di CloudTrail

Anda dapat menggunakan CloudTrail untuk melihat permintaan yang dibuat untuk mengambil peran atau menggabungkan pengguna. Anda juga dapat melihat peran atau permintaan pengguna untuk mengambil tindakan dalam AWS. File CloudTrail log menyertakan informasi tentang identitas sumber yang ditetapkan untuk peran yang diasumsikan atau sesi pengguna federasi. Untuk informasi selengkapnya, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail

Misalnya, asumsikan bahwa pengguna membuat AWS STS `AssumeRole` permintaan, dan menetapkan identitas sumber. Anda dapat menemukan `sourceIdentity` informasi di `requestParameters` kunci di CloudTrail log Anda.

Example Contoh bagian requestParameters dalam AWS CloudTrail log

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Jika pengguna menggunakan sesi peran yang diasumsikan untuk melakukan tindakan, informasi identitas sumber hadir dalam `userIdentity` kunci di CloudTrail log.

Example Contoh kunci userIdentity dalam AWS CloudTrail log

```
{
  "eventVersion": "1.08",
  "userIdentity": {
```

```
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Untuk melihat contoh peristiwa AWS STS API di CloudTrail log, lihatContoh peristiwa IAM API di log CloudTrail. Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat CloudTrail Referensi Peristiwa di Panduan AWS CloudTrail Pengguna.

Izin untuk GetFederationToken

Operasi `GetFederationToken` dipanggil oleh pengguna IAM dan mengembalikan kredensial sementara untuk pengguna tersebut. Operasi ini menggabungkan pengguna. Izin yang ditetapkan untuk sesi pengguna AWS STS federasi didefinisikan di salah satu dari dua tempat:

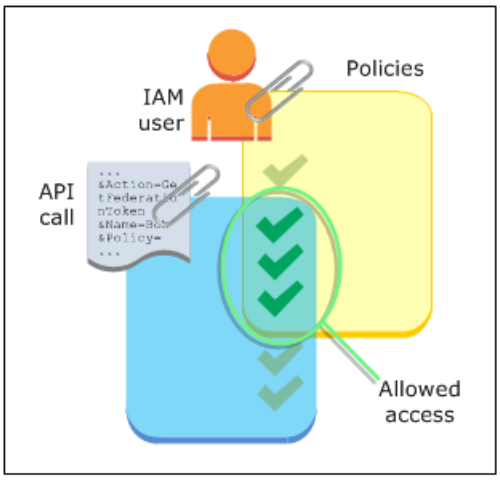
- Kebijakan sesi lewat sebagai sebuah parameter dari Panggilan API `GetFederationToken`. (Ini paling umum.)
- Kebijakan berbasis sumber daya yang secara eksplisit menamai sesi pengguna AWS STS federasi dalam elemen kebijakan. `Principal` (Ini lebih jarang terjadi.)

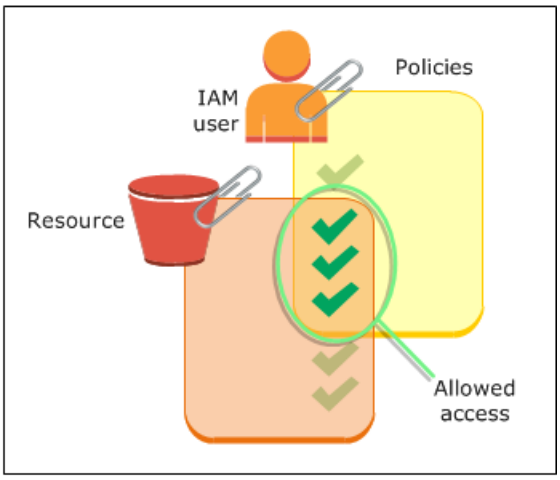
Kebijakan sesi adalah kebijakan lanjutan yang Anda jalankan sebagai parameter saat Anda secara terprogram membuat sebuah sesi sementara. Saat Anda membuat sesi pengguna AWS STS gabungan dan meneruskan kebijakan sesi, izin sesi yang dihasilkan adalah persimpangan kebijakan

berbasis identitas pengguna dan kebijakan sesi. Anda tidak dapat menggunakan kebijakan sesi untuk memberikan lebih banyak izin daripada yang diizinkan oleh kebijakan berbasis identitas dari pengguna yang sedang digabung.

Dalam kebanyakan kasus jika Anda tidak memberikan kebijakan dengan panggilan API `GetFederationToken`, kredensial keamanan sementara yang dihasilkan tidak memiliki izin. Namun, kebijakan berbasis sumber daya dapat memberikan izin tambahan untuk sesi tersebut. Anda dapat mengakses sumber daya dengan sebuah kebijakan berbasis sumber daya yang menentukan sesi Anda sebagai prinsipal yang diizinkan.

Gambar berikut ini menunjukkan gambaran visual tentang cara kebijakan berinteraksi untuk menentukan izin bagi kredensial keamanan sementara yang dikembalikan oleh panggilan ke `GetFederationToken`.





Contoh: Menetapkan izin menggunakan GetFederationToken

Anda dapat menggunakan tindakan API `GetFederationToken` dengan berbagai jenis kebijakan. Berikut adalah beberapa contoh.

Kebijakan yang terlampir pada pengguna IAM

Dalam contoh ini, Anda memiliki aplikasi klien berbasis browser yang mengandalkan dua layanan web backend. Satu layanan backend adalah server autentikasi Anda sendiri yang menggunakan identitas sistem Anda sendiri untuk mengautentikasi aplikasi klien. Layanan backend lainnya adalah layanan AWS yang menyediakan beberapa fungsionalitas aplikasi klien. Aplikasi klien diautentikasi oleh server Anda, dan server Anda membuat atau mengambil kebijakan izin yang sesuai. Server Anda kemudian memanggil API `GetFederationToken` untuk mendapatkan kredensial keamanan sementara, dan mengembalikan kredensial tersebut ke aplikasi klien. Aplikasi klien kemudian dapat

membuat permintaan langsung ke AWS layanan dengan kredenal keamanan sementara. Arsitektur ini memungkinkan aplikasi klien untuk membuat AWS permintaan tanpa menyematkan AWS kredensi jangka panjang.

Server otentikasi Anda memanggil `GetFederationToken` API dengan kredensi keamanan jangka panjang dari pengguna IAM bernama. `token-app` Tetapi kredensi pengguna IAM jangka panjang tetap ada di server Anda dan tidak pernah didistribusikan ke klien. Contoh kebijakan berikut dilampirkan ke pengguna `token-app` IAM dan mendefinisikan kumpulan izin terluas yang diperlukan oleh pengguna AWS STS federasi (klien) Anda. Perhatikan bahwa `sts:GetFederationToken` izin diperlukan untuk layanan otentikasi Anda untuk mendapatkan kredenial keamanan sementara bagi pengguna federasi AWS STS .

Example Contoh kebijakan yang dilampirkan ke aplikasi **`token`** pengguna IAM yang memanggil **`GetFederationToken`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

Kebijakan sebelumnya memberikan beberapa izin kepada pengguna IAM. Namun, kebijakan ini saja tidak memberikan izin apa pun kepada pengguna AWS STS federasi. Jika pengguna IAM ini memanggil `GetFederationToken` dan tidak meneruskan kebijakan sebagai parameter panggilan API, pengguna AWS STS federasi yang dihasilkan tidak memiliki izin yang efektif.

Kebijakan sesi diberikan sebagai parameter

Cara paling umum untuk memastikan bahwa pengguna AWS STS federasi diberi izin yang sesuai adalah dengan meneruskan kebijakan sesi dalam panggilan `GetFederationToken` API. Memperluas contoh sebelumnya, bayangkan yang `GetFederationToken` dipanggil dengan kredensi pengguna IAM. `token-app` Bayangkan sesi kebijakan berikut ini diberikan sebagai parameter panggilan API. Pengguna AWS STS federasi yang dihasilkan memiliki izin untuk mencantumkan konten bucket Amazon S3 yang diberi nama. `productionapp` Pengguna tidak dapat melakukan Amazon S3 `GetObject``PutObject`, dan `DeleteObject` tindakan pada item di bucket. `productionapp`

Pengguna gabungan diberi izin ini karena izin tersebut merupakan pertemuan antara kebijakan pengguna IAM dan kebijakan sesi yang Anda lewati.

Pengguna AWS STS federasi tidak dapat melakukan tindakan di Amazon SNS, Amazon SQS, Amazon DynamoDB, atau di bucket S3 apa pun kecuali. `productionapp` Tindakan ini ditolak meskipun izin tersebut diberikan kepada pengguna IAM yang terkait dengan panggilan tersebut`GetFederationToken`.

Example Contoh kebijakan sesi lewat sebagai sebuah parameter dari Panggilan API **`GetFederationToken`**.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Resource-based kebijakan

Beberapa AWS sumber daya mendukung kebijakan berbasis sumber daya, dan kebijakan ini menyediakan mekanisme lain untuk memberikan izin langsung kepada pengguna federasi. AWS STS Hanya beberapa AWS layanan yang mendukung kebijakan berbasis sumber daya. Misalnya, Amazon S3 memiliki bucket, Amazon SNS memiliki topik, dan Amazon SQS memiliki antrean yang dapat Anda lampirkan kebijakannya. Untuk daftar semua layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM dan tinjau kolom "Resource-basedkebijakan" pada tabel. Anda dapat menggunakan kebijakan berbasis sumber daya untuk menetapkan izin langsung ke pengguna federasi. AWS STS Lakukan ini dengan menentukan Nama Sumber Daya Amazon (ARN) pengguna AWS STS federasi dalam `Principal` elemen kebijakan berbasis sumber daya. Contoh berikut menggambarkan hal ini dan memperluas contoh sebelumnya, menggunakan bucket S3 bernama `productionapp`.

Kebijakan berbasis sumber daya berikut ini terlampir di bucket. Kebijakan bucket ini memungkinkan pengguna AWS STS federasi bernama Carol untuk mengakses bucket. Ketika kebijakan contoh yang dijelaskan sebelumnya dilampirkan ke pengguna `token-app` IAM, pengguna AWS STS federasi bernama Carol memiliki izin untuk melakukan`s3:GetObject`,`s3:PutObject`, dan `s3:DeleteObject` tindakan pada bucket bernama. `productionapp` Hal ini berlaku bahkan ketika kebijakan sesi tidak diberikan sebagai parameter panggilan API `GetFederationToken`. Itu karena dalam kasus ini pengguna AWS STS federasi bernama Carol telah secara eksplisit diberikan izin oleh kebijakan berbasis sumber daya berikut.

Ingat, pengguna AWS STS federasi diberikan izin hanya jika izin tersebut secara eksplisit diberikan kepada pengguna IAM dan pengguna federasi. AWS STS Mereka juga dapat diberikan (dalam akun) oleh kebijakan berbasis sumber daya yang secara eksplisit menyebutkan nama pengguna AWS STS federasi dalam `Principal` elemen kebijakan, seperti pada contoh berikut.

Example Contoh kebijakan bucket yang mengijinkan akses ke pengguna gabungan

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Principal": {
            "AWS": "arn:aws:sts::111122223333:federated-user/Carol"
        },
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": [
            "arn:aws:s3:::productionapp/*"
        ]
    }
}
```

Untuk informasi selengkapnya tentang bagaimana kebijakan dievaluasi, lihat Logika evaluasi kebijakan.

Izin untuk GetSessionToken

Alasan utama untuk menghubungi operasi API `GetSessionToken` atau Perintah CLI `get-session-token` adalah ketika pengguna harus diautentikasi dengan autentikasi Autentikasi Multi-Faktor (MFA). Hal tersebut memungkinkan untuk menulis kebijakan yang mengizinkan tindakan tertentu hanya jika tindakan tersebut diminta oleh pengguna yang telah diautentikasi dengan MFA. Agar berhasil melewati pemeriksaan otorisasi MFA, pengguna harus melakukan panggilan pertama `GetSessionToken` dan menyertakan opsional parameter `SerialNumber` dan `TokenCode`. Jika pengguna berhasil mengautentikasi dengan sebuah perangkat MFA, kredensial dikembalikan

oleh operasi API `GetSessionToken` mencakup konteks MFA. Konteks ini menunjukkan bahwa pengguna diautentikasi dengan MFA dan diotorisasi untuk operasi API yang memerlukan autentikasi MFA.

Izin diperlukan untuk GetSessionToken

Tidak ada izin yang diperlukan bagi pengguna untuk mendapatkan token sesi. Tujuan dari operasi `GetSessionToken` adalah mengautentikasi pengguna menggunakan MFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol operasi autentikasi.

Untuk memberikan izin untuk melakukan sebagian besar AWS operasi, Anda menambahkan tindakan dengan nama yang sama ke kebijakan. Misalnya, untuk membuat pengguna, Anda harus menggunakan operasi API `CreateUser`, perintah CLI `create-user`, atau Konsol Manajemen AWS. Untuk melakukan operasi ini, Anda harus memiliki kebijakan yang mengijinkan Anda untuk mengakses tindakan `CreateUser`.

JSON

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

Anda dapat memasukkan tindakan `GetSessionToken` dalam kebijakan Anda, tetapi tidak itu berpengaruh pada kemampuan pengguna untuk melakukan operasi `GetSessionToken`.

Izin yang diberikan oleh GetSessionToken

Jika `GetSessionToken` yang dipanggil dengan kredensial pengguna IAM, kredensial keamanan sementara memiliki izin yang sama dengan pengguna IAM. Demikian pula, jika `GetSessionToken` dipanggil dengan Pengguna root akun AWS kredenal, kredenal keamanan sementara memiliki izin pengguna root.

> **Note**
>
> Kami menyarankan agar Anda tidak memanggil `GetSessionToken` dengan kredensial pengguna root. Sebaliknya, ikuti praktik terbaik kami dan membuat pengguna IAM dengan izin yang mereka butuhkan. Kemudian gunakan pengguna IAM ini untuk interaksi sehari-hari dengan AWS.

Kredensial sementara yang Anda dapatkan ketika Anda memanggil `GetSessionToken` memiliki kemampuan dan batasan sebagai berikut:

- Anda dapat menggunakan kredensi untuk mengakses Konsol Manajemen AWS dengan meneruskan kredensialnya ke titik akhir masuk tunggal federasi di. `https://signin.aws.amazon.com/federation` Untuk informasi selengkapnya, lihat Aktifkan akses broker identitas khusus ke AWS konsol.
- Anda tidak dapat menggunakan kredensialnya untuk memanggil operasi IAM atau AWS STS API. Anda dapat menggunakannya untuk memanggil operasi API untuk AWS layanan lain.

Bandingkan operasi API ini serta batasan dan kemampuannya dengan operasi API lain yang membuat kredensial keamanan sementara di Bandingkan AWS STS kredensialnya

Untuk informasi selengkapnya tentang akses API yang dilindungi MFA menggunakan `GetSessionToken`, lihat Akses API aman dengan MFA.

Menonaktifkan izin untuk kredensial keamanan sementara

Kredensi keamanan sementara berlaku hingga kedaluwarsa. Kredensi ini berlaku untuk durasi yang ditentukan, dari 900 detik (15 menit) hingga maksimum 129.600 detik (36 jam). Durasi sesi default adalah 43.200 detik (12 jam). Anda dapat mencabut kredensil ini, tetapi Anda juga harus mengubah izin untuk pengguna IAM atau peran untuk menghentikan penggunaan kredensil yang disusupi untuk aktivitas akun berbahaya. Izin yang ditetapkan untuk kredensil keamanan sementara dievaluasi setiap kali digunakan untuk membuat permintaan. AWS Setelah Anda menghapus semua izin dari kredensi, AWS permintaan yang menggunakannya gagal.

Mungkin perlu beberapa menit agar pembaruan kebijakan diterapkan. Untuk sesi peran IAM, Anda dapat mencabut kredensil keamanan sementara peran tersebut untuk memaksa semua pengguna yang mengasumsikan peran tersebut untuk mengautentikasi ulang dan meminta kredensil baru. Untuk informasi selengkapnya, lihat Mencabut kredenal keamanan sementara peran tersebut.

Anda tidak dapat mengubah izin untuk file Pengguna root akun AWS. Demikian juga, Anda tidak dapat mengubah izin untuk kredensial keamanan sementara yang dibuat dengan memanggil `GetFederationToken` atau `GetSessionToken` saat masuk sebagai pengguna root. Karena alasan ini, kami menyarankan agar Anda tidak memanggil `GetFederationToken` atau `GetSessionToken` sebagai pengguna root.

Untuk prosedur tentang cara mengubah izin untuk pengguna IAM, lihat. Mengubah izin untuk pengguna IAM

Untuk prosedur tentang cara mengubah izin untuk peran IAM, lihat. Memperbarui izin untuk peran

> **Important**
>
> Anda tidak dapat mengedit peran di IAM yang dibuat dari kumpulan izin Pusat Identitas IAM. Anda harus mencabut sesi set izin aktif untuk pengguna di Pusat Identitas IAM. Untuk informasi selengkapnya, lihat Mencabut sesi peran IAM aktif yang dibuat oleh set izin di Panduan Pengguna Pusat Identitas IAM.

Topik

- Tolak akses ke semua sesi peran IAM yang terkait dengan peran
- Tolak akses ke sesi peran IAM tertentu
- Tolak akses ke sesi kredensi keamanan sementara dengan kunci konteks kondisi
- Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya

Tolak akses ke semua sesi peran IAM yang terkait dengan peran

Prosedur ini menolak izin untuk semua sesi peran IAM yang terkait dengan peran. Gunakan pendekatan ini ketika Anda khawatir tentang akses yang mencurigakan dengan:

- Prinsipal dari akun lain menggunakan akses lintas akun
- Identitas pengguna eksternal dengan izin untuk mengakses AWS sumber daya di akun Anda
- Pengguna yang telah diautentikasi dalam aplikasi seluler atau web dengan penyedia OIDC

Untuk mengubah atau menghapus izin yang ditetapkan ke kredensil keamanan sementara yang diperoleh dengan memanggil`AssumeRole`,, atau,`AssumeRoleWithSAML`, atau

`AssumeRoleWithWebIdentity`GetFederationToken, atau`GetSessionToken`, Anda dapat mengedit atau menghapus kebijakan berbasis identitas yang menentukan izin untuk peran tersebut.

> **Important**
>
> Jika ada kebijakan berbasis sumber daya yang memungkinkan akses utama, Anda juga harus menambahkan penolakan eksplisit untuk sumber daya tersebut. Lihat Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya untuk detail.

Untuk menolak akses ke semua sesi peran IAM yang terkait dengan peran

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM.
2. Di panel navigasi, pilih Peran..
3. Pilih nama peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
4. Pilih tab Izin.
5. Pilih kebijakan yang relevan untuk diedit. Sebelum Anda mengedit kebijakan terkelola pelanggan, tinjau tab Entitas yang dilampirkan untuk menghindari gangguan akses ke identitas lain yang mungkin memiliki kebijakan yang sama.
6. Pilih tab JSON dan perbarui kebijakan untuk menolak semua sumber daya dan tindakan.

   > **Note**
   >
   > Izin ini sama dengan yang ada di kebijakan AWS terkelola AWSDenySemua. Anda dapat melampirkan kebijakan AWS terkelola ini ke setiap pengguna IAM atau peran yang ingin Anda tolak semua aksesnya.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAll",
            "Effect": "Deny",
```

```
            "Action": [
                "*"
            ],
            "Resource": "*"
        }
    ]
}
```

7. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Saat memperbarui kebijakan, perubahan akan memengaruhi izin semua kredensil keamanan sementara yang terkait dengan peran tersebut, termasuk kredensil yang dikeluarkan sebelum Anda mengubah kebijakan izin peran.

Setelah memperbarui kebijakan, Anda dapat mencabut kredensil keamanan sementara peran tersebut untuk segera mencabut semua izin atas kredenal yang dikeluarkan peran tersebut.

Tolak akses ke sesi peran IAM tertentu

Saat Anda memperbarui peran IAM dengan kebijakan penolakan semua atau menghapus peran sepenuhnya, semua pengguna yang memiliki akses ke peran tersebut akan terganggu. Anda dapat menolak akses tanpa memengaruhi izin semua sesi lain yang terkait dengan peran tersebut.

Izin `Principal` dapat ditolak menggunakan kunci konteks kondisi atau kebijakan berbasis sumber daya.

> **Tip**
>
> Anda dapat menemukan pengguna ARNs federasi menggunakan AWS CloudTrail log. Untuk informasi selengkapnya, lihat Cara Mudah Mengidentifikasi Pengguna Federasi Anda dengan Menggunakan AWS CloudTrail.

Tolak akses ke sesi kredensi keamanan sementara dengan kunci konteks kondisi

Anda dapat menggunakan kunci konteks kondisi dalam kebijakan berbasis identitas dalam situasi di mana Anda ingin menolak akses ke sesi kredensi keamanan sementara tertentu tanpa memengaruhi izin pengguna IAM atau peran yang membuat kredensialnya. Untuk peran IAM, setelah memperbarui kebijakan, Anda juga dapat mencabut sesi kredensil keamanan sementara peran tersebut untuk segera mencabut semua kredensil yang dikeluarkan.

Untuk informasi selengkapnya tentang kunci konteks kondisi, lihatAWS kunci konteks kondisi global.

aws: PrincipalArn

Anda dapat menggunakan kunci konteks kondisi aws: PrincipalArn dalam kebijakan berbasis identitas untuk menolak akses ke prinsipal tertentu dengan Nama Sumber Daya Amazon (ARN) mereka. Anda melakukannya dengan menentukan ARN pengguna IAM, peran, AWS STS atau sesi pengguna gabungan yang terkait dengan kredensil keamanan sementara dalam elemen Kondisi kebijakan.

Untuk menolak akses ke kepala sekolah tertentu oleh ARN mereka

1. Di panel navigasi konsol IAM, pilih Pengguna atau Peran.
2. Pilih nama pengguna IAM atau peran yang akan diedit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar.
3. Pilih tab Izin.
4. Pilih kebijakan yang relevan untuk diedit. Sebelum Anda mengedit kebijakan terkelola pelanggan, tinjau tab Entitas yang dilampirkan untuk menghindari gangguan akses ke identitas lain yang mungkin memiliki kebijakan yang sama.
5. Pilih tab JSON dan tambahkan pernyataan penolakan untuk ARN utama seperti yang ditunjukkan pada contoh berikut.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:iam::222222222222:federated-user/USERNAME"
          ]
        }
      }
```

```
    }
  ]
}
```

6. Di halaman Tinjauan, tinjau Ringkasan kebijakan lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

aws: SourceIdentity

Anda dapat menggunakan kunci konteks kondisi aws: SourceIdentity dalam kebijakan berbasis identitas untuk menolak akses ke identitas sumber tertentu yang terkait dengan sesi peran IAM. Ini berlaku selama sesi peran dikeluarkan dengan menyetel parameter `SourceIdentity` permintaan saat prinsipal mengambil peran menggunakan AWS STS `assume-role` perintah* CLI apa pun, atau AWS STS `AssumeRole` operasi* API. Anda melakukannya dengan menentukan identitas sumber yang terkait dengan kredensi keamanan sementara dalam `Condition` elemen kebijakan.

Tidak seperti kunci konteks sts:RoleSessionName, setelah identitas sumber disetel, nilainya tidak dapat diubah. `aws:SourceIdentity`Kuncinya ada dalam konteks permintaan untuk semua tindakan yang diambil oleh peran. Identitas sumber tetap ada di sesi peran berikutnya saat Anda menggunakan kredenal sesi untuk mengambil peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Kebijakan berikut menunjukkan contoh bagaimana Anda dapat menolak akses ke sesi kredensi keamanan sementara menggunakan kunci `aws:SourceIdentity` konteks kondisi. Jika Anda menentukan identitas sumber yang terkait dengan sesi peran, itu akan menolak sesi peran dengan identitas sumber bernama tanpa memengaruhi izin peran yang membuat kredensil. Untuk contoh ini, identitas sumber yang ditetapkan oleh kepala sekolah saat sesi peran dikeluarkan adalah`nikki_wolf@example.com`. Setiap permintaan yang dibuat oleh sesi peran dengan identitas sumber `nikki_wolf@example.com` akan ditolak karena identitas sumber disertakan dalam kondisi kebijakan dan Efek kebijakan disetel ke`Deny`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
```

```
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": [
            "nikki_wolf@example.com",
            "<source identity value>"
          ]
        }
      }
    }
  ]
}
```

aws:userid

Anda dapat menggunakan kunci konteks kondisi aws:userid dalam kebijakan berbasis identitas untuk menolak akses ke semua atau sesi kredensi keamanan sementara tertentu yang terkait dengan pengguna atau peran IAM. Anda melakukannya dengan menentukan pengenal unik (ID) pengguna IAM, peran, atau sesi pengguna AWS STS federasi yang terkait dengan kredensil keamanan sementara dalam elemen kebijakan. `Condition`

Kebijakan berikut menunjukkan contoh bagaimana Anda dapat menolak akses ke sesi kredensi keamanan sementara menggunakan kunci `aws:userid` konteks kondisi.

- `AIDAXUSER1`mewakili ID unik untuk pengguna IAM. Menentukan ID unik pengguna IAM sebagai nilai untuk kunci konteks `aws:userid` akan menolak akses ke pengguna IAM. Ini termasuk sesi kredensi keamanan sementara yang dibuat dengan memanggil `GetSessionToken` API.
- `AROAXROLE1:*`mewakili ID unik untuk semua sesi yang terkait dengan peran IAM. Menentukan ID unik peran IAM dan karakter wildcard (*) di bagian caller-specified-role-session -name sebagai nilai untuk kunci konteks `aws:userid` akan menolak semua sesi yang terkait dengan peran tersebut.
- `AROAXROLE2:<caller-specified-role-session-name>`mewakili ID unik untuk sesi peran yang diasumsikan. Di bagian caller-specified-role-session -name dari ID unik peran yang diasumsikan, Anda dapat menentukan nama sesi peran atau karakter wildcard jika operator kondisi digunakan. StringLike Jika Anda menentukan nama sesi peran, itu akan menolak sesi peran bernama tanpa memengaruhi izin peran yang membuat kredensialnya. Jika Anda menentukan karakter wildcard untuk nama sesi peran, itu akan menolak semua sesi yang terkait dengan peran tersebut.

> **Note**
>
> Nama sesi peran yang ditentukan pemanggil, yang merupakan bagian dari pengenal unik untuk sesi peran yang diasumsikan, dapat berubah selama rantai peran. Role chaining terjadi ketika satu peran mengambil peran lain. Nama sesi peran disetel menggunakan parameter `RoleSessionName` permintaan saat prinsipal mengasumsikan peran menggunakan operasi AWS STS `AssumeRole` API.

- `account-id:<federated-user-caller-specified-name>`mewakili ID unik untuk sesi pengguna AWS STS federasi. Pengguna IAM membuat sesi ini dengan memanggil `GetFederationToken` API. Menentukan ID unik untuk sesi pengguna AWS STS federasi menyangkal sesi federasi bernama tanpa memengaruhi izin pengguna IAM yang membuat kredensialnya.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1:*",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "123456789012:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Untuk contoh spesifik dari nilai kunci utama, lihatNilai-nilai kunci utama. Untuk informasi tentang pengidentifikasi unik IAM dan cara mendapatkannya, lihat. Pengidentifikasi unik

Tolak akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya

Untuk membatasi akses ke prinsipal tertentu dengan kebijakan berbasis sumber daya, Anda dapat menggunakan kunci aws: PrincipalArn konteks kondisi atau dalam elemen. aws: SourceIdentity `Condition` Kebijakan berbasis sumber daya adalah kebijakan izin yang dilampirkan pada sumber daya dan kontrol siapa yang dapat mengakses sumber daya dan tindakan apa yang dapat mereka lakukan terhadapnya.

Bila Anda menggunakan kunci `aws:PrincipalARN` konteks, tentukan ARN pengguna IAM, peran, atau sesi pengguna AWS STS federasi yang terkait dengan kredensil keamanan sementara dalam elemen Kondisi kebijakan. Contoh kebijakan berikut menunjukkan cara menggunakan kunci `aws:PrincipalARN` konteks dalam kebijakan berbasis sumber daya:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "ArnEquals": {
        "aws:PrincipalArn": [
          "arn:aws:iam::222222222222:role/ROLENAME",
          "arn:aws:iam::222222222222:user/USERNAME",
          "arn:aws:sts::222222222222:federated-user/USERNAME"
        ]
      }
    }
  }
}
```

Bila Anda menggunakan kunci `aws:SourceIdentity` konteks, tentukan nilai identitas sumber yang terkait dengan kredensil keamanan sementara peran dalam `Condition` elemen kebijakan. Ini berlaku selama sesi peran dikeluarkan dengan menyetel parameter `SourceIdentity` permintaan

saat prinsipal mengambil peran menggunakan AWS STS `assume-role` perintah* CLI apa pun, atau AWS STS `AssumeRole` operasi* API. Contoh berikut menunjukkan cara menggunakan kunci `aws:SourceIdentity` konteks dalam kebijakan berbasis sumber daya:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "StringLike": {
        "aws:SourceIdentity": [
          "nikki_wolf@example.com",
          "<source identity value>"
        ]
      }
    }
  }
}
```

Jika Anda hanya memperbarui kebijakan berbasis identitas untuk prinsipal, mereka masih dapat melakukan tindakan yang diizinkan dalam kebijakan berbasis sumber daya, kecuali jika tindakan tersebut secara eksplisit ditolak dalam kebijakan berbasis identitas.

Untuk menolak akses ke prinsipal tertentu dalam kebijakan berbasis sumber daya

1. Lihat AWS layanan yang bekerja dengan IAM untuk melihat apakah layanan mendukung kebijakan berbasis sumber daya.
2. Masuk ke Konsol Manajemen AWS dan buka konsol untuk layanan ini. Setiap layanan memiliki lokasi yang berbeda di konsol untuk melampirkan kebijakan.
3. Edit kebijakan berbasis sumber daya. Tambahkan pernyataan kebijakan penolakan untuk menentukan informasi identifikasi kredensi:

    a. Dalam `Principal` elemen, masukkan wildcard (*). Kepala sekolah akan dibatasi dalam `Condition` elemen.

b. Dalam `Effect` elemen, masukkan “Deny.”

c. Masuk`Action`, masukkan namespace layanan dan nama tindakan yang akan ditolak. Untuk menolak semua tindakan, gunakan karakter wildcard (*). Sebagai contoh: `"s3:*"`.

d. Dalam `Resource` elemen, masukkan ARN dari sumber daya target. Sebagai contoh: `"arn:aws:s3:::amzn-s3-demo-bucket"`.

e. Dalam `Condition` elemen, tentukan kunci `aws:PrincipalARN` atau `aws:SourceIdentity` konteks.

   Jika Anda menggunakan tombol `aws:PrincipalARN` konteks, masukkan ARN kepala sekolah untuk menolak akses.

   Jika Anda menggunakan kunci `aws:SourceIdentity` konteks, masukkan nilai identitas sumber yang ditetapkan dalam sesi peran untuk menolak akses.

4. Simpan pekerjaan Anda.

Memberikan izin untuk membuat kredensial keamanan sementara

Secara default, pengguna IAM tidak memiliki izin untuk membuat kredensil keamanan sementara untuk sesi dan peran pengguna AWS STS federasi. Anda harus menggunakan kebijakan untuk memberikan izin ini kepada pengguna Anda. Meskipun Anda dapat memberikan izin secara langsung kepada pengguna, kami sangat menyarankan agar Anda memberikan izin kepada grup. Ini menjadikan pengelolaan izin jauh lebih mudah. Saat seseorang tidak lagi perlu melakukan tugas yang terkait dengan izin, Anda cukup menghapusnya dari grup. Jika orang lain perlu melakukan tugas tersebut, tambahkan mereka ke grup untuk memberikan izin.

Untuk memberikan izin grup IAM untuk membuat kredensil keamanan sementara untuk sesi atau peran pengguna AWS STS gabungan, Anda melampirkan kebijakan yang memberikan salah satu atau kedua hak istimewa berikut:

- Untuk kepala sekolah federasi OIDC dan SALL untuk mengakses peran IAM, berikan akses ke. AWS STS `AssumeRole`
- Untuk pengguna AWS STS federasi yang tidak memerlukan peran, berikan akses ke AWS STS `GetFederationToken`.

Untuk informasi selengkapnya tentang perbedaan antara operasi API `AssumeRole` dan `GetFederationToken`, lihat Minta kredentif keamanan sementara.

Pengguna IAM juga dapat memanggil `GetSessionToken` untuk membuat kredensial keamanan sementara. Tidak ada izin yang diperlukan bagi pengguna untuk memanggil `GetSessionToken`. Tujuan dari operasi ini adalah mengautentikasi pengguna menggunakan MFA. Anda tidak dapat menggunakan kebijakan untuk mengontrol autentikasi. Ini artinya Anda tidak dapat mencegah pengguna IAM melakukan panggilan `GetSessionToken` untuk membuat kredensial sementara.

Example Contoh kebijakan yang memberikan izin untuk mengambil peran

Contoh kebijakan berikut memberikan izin `AssumeRole` untuk memanggil `UpdateApp` peran dalam Akun AWS `123123123123`. Saat `AssumeRole` digunakan, pengguna (atau aplikasi) yang membuat kredensial keamanan atas nama pengguna gabungan tidak dapat mengurangi izin apapun yang belum ditentukan dalam kebijakan izin peran.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Contoh kebijakan yang memberikan izin untuk membuat kredensial keamanan sementara bagi pengguna gabungan.

Contoh kebijakan berikut memberikan izin untuk mengakses `GetFederationToken`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Saat Anda memberi izin kepada pengguna IAM untuk membuat kredensil keamanan sementara untuk pengguna AWS STS federasi`GetFederationToken`, ketahuilah bahwa ini memungkinkan pengguna tersebut untuk mendelegasikan izin mereka sendiri. Untuk informasi selengkapnya tentang mendelegasikan izin di seluruh pengguna IAM dan Akun AWS, lihat. Contoh kebijakan untuk mendelegasikan akses Untuk informasi selengkapnya tentang mengatur izin pada kredensial keamanan sementara, lihat Izin untuk kredensial keamanan sementara.

Example Contoh kebijakan yang memberikan izin terbatas kepada pengguna untuk membuat kredensial keamanan sementara bagi pengguna gabungan.

Saat Anda mengizinkan pengguna IAM memanggil `GetFederationToken`, ini adalah penerapan terbaik untuk membatasi izin yang dapat dikeluarkan oleh pengguna IAM. Misalnya, kebijakan berikut menunjukkan cara mengizinkan pengguna IAM membuat kredensil keamanan sementara hanya untuk pengguna AWS STS federasi yang namanya dimulai dengan Manajer.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Memberikan izin untuk menggunakan sesi konsol yang disempurnakan identitas

Sesi konsol yang disempurnakan identitas memungkinkan AWS IAM Identity Center pengguna dan sesi IDs untuk disertakan dalam sesi AWS konsol pengguna saat mereka masuk. Misalnya, Amazon Q Developer Pro menggunakan sesi konsol yang disempurnakan identitas untuk mempersonalisasi pengalaman layanan. Untuk informasi selengkapnya tentang sesi konsol yang disempurnakan identitas, lihat Mengaktifkan sesi konsol yang disempurnakan identitas di Panduan Pengguna.AWS

IAM Identity Center Untuk informasi tentang penyiapan Pengembang Amazon Q, lihat Menyiapkan Pengembang Amazon Q di Panduan Pengguna Pengembang Amazon Q.

Agar sesi konsol yang disempurnakan identitas tersedia bagi pengguna, Anda harus menggunakan kebijakan berbasis identitas untuk memberikan `sts:SetContext` izin kepada prinsipal IAM untuk sumber daya yang mewakili sesi konsol mereka sendiri.

> **Important**
>
> Secara default, pengguna tidak memiliki izin untuk menyetel konteks untuk sesi konsol yang disempurnakan identitas mereka. Untuk mengizinkan hal ini, Anda harus memberikan `sts:SetContext` izin kepada kepala IAM dalam kebijakan berbasis identitas seperti yang ditunjukkan pada contoh kebijakan di bawah ini.

Contoh kebijakan berbasis identitas berikut memberikan `sts:SetContext` izin kepada prinsipal IAM, memungkinkan prinsipal untuk mengatur konteks sesi konsol yang ditingkatkan identitas untuk sesi konsol mereka sendiri. AWS Sumber daya kebijakan,`arn:aws:sts::`*`account-id`*`:self`, mewakili AWS sesi pemanggil. Segmen `account-id` ARN dapat diganti dengan karakter * wildcard jika kebijakan izin yang sama diterapkan di beberapa akun, seperti saat kebijakan ini diterapkan menggunakan set izin Pusat Identitas IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:SetContext",
            "Resource": "arn:aws:sts::111122223333:self"
        }
    ]
}
```

Kelola AWS STS dalam sebuah Wilayah AWS

Titik akhir regional adalah URL titik masuk dalam wilayah tertentu untuk layanan AWS web. AWS merekomendasikan penggunaan titik akhir Regional AWS Security Token Service (AWS STS) alih-alih titik akhir global untuk mengurangi latensi, membangun redundansi, dan meningkatkan validitas token sesi.

Meskipun titik akhir global (lama) AWS STS `https://sts.amazonaws.com` sangat tersedia, titik akhir ini dihosting di satu Wil AWS ayah, AS Timur (Virginia Utara), dan seperti titik akhir lainnya, titik akhir tidak menyediakan failover otomatis ke titik akhir di Wilayah lain.

- Kurangi latensi — Dengan melakukan AWS STS panggilan ke titik akhir yang secara geografis lebih dekat dengan layanan dan aplikasi Anda, Anda dapat mengakses AWS STS layanan dengan latensi yang lebih rendah dan waktu respons yang lebih baik.
- Membangun redundansi — Anda dapat membatasi efek kegagalan dalam beban kerja ke sejumlah komponen terbatas dengan cakupan penahanan dampak yang dapat diprediksi. Menggunakan AWS STS titik akhir regional memungkinkan Anda menyelaraskan cakupan komponen Anda dengan cakupan token sesi Anda. Untuk informasi selengkapnya tentang pilar keandalan ini, lihat Menggunakan isolasi kesalahan untuk melindungi beban kerja Anda di Ker AWS Well-Architected angka Kerja.
- Tingkatkan validitas token sesi — Token sesi dari AWS STS titik akhir Regional valid di semua Wilayah AWS. Token sesi dari titik akhir STS global hanya valid Wilayah AWS jika diaktifkan secara default. Jika Anda ingin mengaktifkan Wilayah baru untuk akun Anda, Anda dapat menggunakan token sesi dari AWS STS titik akhir Regional. Jika Anda memilih untuk menggunakan titik akhir global, Anda harus mengubah kompatibilitas Wilayah token AWS STS sesi untuk titik akhir global. Melakukannya memastikan bahwa token valid di semua Wilayah AWS.

Untuk daftar Wil AWS STS ayah dan titik akhir mereka, lihatAWS STS Wilayah dan titik akhir.

> **Note**
>
> AWS telah membuat perubahan pada AWS Security Token Service (AWS STS) titik akhir global (`https://sts.amazonaws.com`) di Wilayah yang diaktifkan secara default untuk meningkatkan ketahanan dan kinerjanya. AWS STS permintaan ke titik akhir global secara otomatis dilayani Wilayah AWS sama dengan beban kerja Anda. Perubahan ini tidak akan diterapkan ke Wilayah pilihan. Kami menyarankan Anda menggunakan titik akhir AWS STS

regional yang sesuai. Untuk informasi selengkapnya, lihat AWS STS perubahan titik akhir global.

Topik

- Mengaktifkan dan menonaktifkan AWS STS dalam sebuah Wilayah AWS
- Menulis kode untuk digunakan AWS STS Wilayah
- Mengelola token sesi titik akhir global

Mengaktifkan dan menonaktifkan AWS STS dalam sebuah Wilayah AWS

Saat Anda mengaktifkan AWS STS titik akhir untuk Wilayah, AWS STS dapat mengeluarkan kredenSIAL sementara kepada pengguna dan peran di akun Anda yang membuat AWS STS permintaan. Kredensial tersebut kemudian dapat digunakan di setiap Wilayah yang diaktifkan secara default atau diaktifkan secara manual.

Untuk Wilayah yang diaktifkan secara default, Anda harus mengaktifkan AWS STS titik akhir Regional di akun tempat kredenSIAL sementara dibuat. Tidak masalah apakah pengguna masuk ke akun yang sama atau akun yang berbeda saat mereka mengajukan permintaan.

Saat meminta kredenSIAL sementara untuk peran di peran lain Akun AWS menggunakan Wilayah yang diaktifkan secara manual, akun target (akun yang berisi peran) harus mengaktifkan Wilayah tersebut untuk AWS STS operasi. Ini memastikan bahwa kredenSIAL keamanan sementara dapat dibuat dengan benar.

Misalnya, bayangkan pengguna di akun A ingin mengirim permintaan `sts:AssumeRole` API ke titik akhir AWS STS Regional `https://sts.ap-southeast-3.amazonaws.com`. Permintaan ini untuk kredenSIAL sementara untuk peran yang dinam `Developer` ai di akun B. Karena permintaannya adalah untuk membuat kredenSIAL untuk entitas di akun B, akun B harus mengaktifkan `ap-southeast-3` Wilayah. Pengguna dari akun A (atau akun lain) dapat memanggil titik akhir `ap-southeast-3` AWS STS untuk meminta kredensial bagi akun B apakah Wilayah diaktifkan di akun mereka atau tidak. Untuk mempelajari selengkapnya, lihat Mengaktifkan atau menonaktifkan Wilayah AWS di akun Anda.

> **Note**
>
> Wilayah Aktif tersedia bagi semua orang yang menggunakan kredensial sementara dalam akun tersebut. Untuk mengontrol pengguna atau peran IAM mana yang dapat mengakses Wilayah, gunakan kunci ketentuan `aws:RequestedRegion` dalam kebijakan izin Anda.

Untuk mengaktifkan atau menonaktifkan AWS STS di Wilayah yang diaktifkan secara default (konsol)

1. Masuk sebagai pengguna root atau pengguna dengan izin untuk melakukan tugas administrasi IAM.
2. Membuka Konsol IAM dan di panel navigasi pilih Pengaturan akun.
3. Di bagian Titik Akhir Layanan Token Keamanan (STS), cari Wilayah yang ingin Anda konfigurasi, lalu pilih Aktif atau Tidak Aktif di kolom status STS.
4. Pada kotak dialog yang terbuka, pilih Aktifkan atau Non aktifkan.

Untuk Wilayah yang harus diaktifkan, kami mengaktifkan AWS STS secara otomatis saat Anda mengaktifkan Wilayah. Setelah Anda mengaktifkan Wilayah, AWS STS selalu aktif untuk Wilayah dan Anda tidak dapat menonaktifkannya. Untuk mempelajari cara mengaktifkan Wilayah yang dinonaktifkan secara default, lihat Menentukan Wilayah AWS akun mana yang dapat digunakan dalam Panduan AWS Account Management Referensi.

Menulis kode untuk digunakan AWS STS Wilayah

Setelah mengaktifkan Wilayah, Anda dapat mengarahkan panggilan AWS STS API ke Wilayah tersebut. Cuplikan kode Java berikut menunjukkan cara mengkonfigurasi `AWSSecurityTokenService` objek untuk membuat permintaan ke Wilayah Eropa (Milan) (eu-south-1).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS merekomendasikan agar Anda melakukan panggilan ke titik akhir Regional. Untuk mempelajari cara mengaktifkan Wilayah secara manual, lihat Men entukan Wilayah AWS akun mana yang dapat digunakan dalam Panduan AWS Account Management Referensi.

Dalam contoh, baris pertama membuat instance `EndpointConfiguration` objek yang dipanggil`regionEndpointConfig`, meneruskan URL titik akhir dan Wilayah AWS sebagai parameter.

Untuk mempelajari cara menyetel titik akhir AWS STS regional menggunakan variabel lingkungan untuk AWS SDK, lihat Titik akhir AWS STS regional di Panduan Referensi AWS SDK dan Alat.

Untuk semua kombinasi bahasa dan lingkungan pemrograman lainnya, mengacu pada dokumentasi untuk SDK terkait.

Mengelola token sesi titik akhir global

Sebagian besar Wilayah AWS diaktifkan untuk operasi di semua Layanan AWS secara default. Wilayah tersebut secara otomatis diaktifkan untuk digunakan dengan AWS STS. Beberapa Wilayah, seperti Asia Pacific (Hong Kong) harus diaktifkan secara manual. Untuk mempelajari selengkapnya tentang mengaktifkan dan menonaktifkan Wilayah AWS, lihat Men Wilayah AWS entukan akun mana yang dapat digunakan di Panduan AWS Account Management Referensi.

Saat Anda mengaktifkan Wil AWS ayah ini, mereka secara otomatis diaktifkan untuk digunakan dengan AWS STS. Anda tidak dapat mengaktifkan titik AWS STS akhir untuk Wilayah yang dinonaktifkan. Token sesi yang valid di semua Wilayah AWS termasuk lebih banyak karakter daripada token yang valid di Wilayah yang diaktifkan secara default. Mengubah pengaturan ini mungkin memengaruhi sistem yang sudah ada di mana Anda menyimpan token untuk sementara waktu.

Anda dapat mengubah pengaturan ini menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API.

Untuk mengubah kesesuaian Wilayah token sesi untuk titik akhir global (konsole)

1. Masuk sebagai pengguna root atau pengguna dengan izin untuk melakukan tugas administrasi IAM. Untuk mengubah kesesuaian token sesi, Anda harus memiliki kebijakan yang memungkinkan tindakan `iam:SetSecurityTokenServicePreferences`.
2. Buka konsol IAM. Di panel navigasi, pilih Pengaturan akun.
3. Di bawah bagian Security Token Service (STS) Token Sesi dari titik akhir STS. T itik akhir global menunjukkan`Valid only in Wilayah AWS enabled by default`. Pilih Ubah.

4. Di kotak dialog U bah kompatibilitas wilayah, pilih Semua Wilayah AWS. Lalu pilih Simpan Perubahan.

> **Note**
>
> Token sesi yang valid di semua Wilayah AWS termasuk lebih banyak karakter daripada token yang valid di Wilayah yang diaktifkan secara default. Mengubah pengaturan ini mungkin memengaruhi sistem yang sudah ada di mana Anda menyimpan token untuk sementara waktu.

Untuk mengubah kompatibilitas Wilayah token sesi untuk titik akhir global (AWS CLI)

Atur versi token sesi. Token versi 1 hanya valid Wilayah AWS yang tersedia secara default. Token ini tidak bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pasifik (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 memuat lebih banyak karakter dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.

- `aws iam set-security-token-service-preferences`

Untuk mengubah kompatibilitas Wilayah token sesi untuk titik akhir global (AWS API)

Atur versi token sesi. Token versi 1 hanya valid Wilayah AWS yang tersedia secara default. Token ini tidak bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pasifik (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 memuat lebih banyak karakter dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.

- `SetSecurityTokenServicePreferences`

AWS STS Wilayah dan titik akhir

> **Note**
>
> AWS telah membuat perubahan pada AWS Security Token Service (AWS STS) titik akhir global (`https://sts.amazonaws.com`) di Wilayah yang diaktifkan secara default untuk meningkatkan ketahanan dan kinerjanya. AWS STS permintaan ke titik akhir global secara otomatis disajikan Wilayah AWS sama dengan beban kerja Anda. Perubahan ini tidak akan diterapkan ke Wilayah keikutsertaan. Kami menyarankan Anda menggunakan titik akhir AWS

STS regional yang sesuai. Untuk informasi selengkapnya, lihat AWS STS perubahan titik akhir global.

Tabel berikut mencantumkan Wilayah dan titik akhirnya. Ini menunjukkan mana yang diaktifkan secara default dan mana yang dapat Anda aktifkan atau nonaktifkan.

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
--Global--	sts.amazonaws.com	✓Ya	✗Tidak
AS Timur (Ohio)	sts.us-east-2.amazonaws.com	✓Ya	✓Ya
AS Timur (Virginia Utara)	sts.us-east-1.amazonaws.com	✓Ya	✗Tidak
AS Barat (California Utara)	sts.us-west-1.amazonaws.com	✓Ya	✓Ya
AS Barat (Oregon)	sts.us-west-2.amazonaws.com	✓Ya	✓Ya
Africa (Cape Town)	sts.af-south-1.amazonaws.com	✗Tidak[1]	✗Tidak
Asia Pasifik (Hong Kong)	sts.ap-east-1.amazonaws.com	✗Tidak[1]	✗Tidak
Asia Pasifik (Hyderabad)	sts.ap-south-2.amazonaws.com	✗Tidak[1]	✗Tidak
Asia Pasifik (Jakarta)	sts.ap-southeast-3.amazonaws.com	✗Tidak[1]	✗Tidak
Asia Pasifik (Malaysia)	sts.ap-southeast-5.amazonaws.com	✗Tidak[1]	✗Tidak

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Asia Pacific (Melbourne)	sts.ap-southeast-4.amazonaws.com	⊗Tidak[1]	⊗Tidak
Asia Pasifik (Mumbai)	sts.ap-south-1.amazonaws.com	⊘Ya	⊘Ya
Asia Pasifik (Osaka)	sts.ap-northeast-3.amazonaws.com	⊘Ya	⊘Ya
Asia Pasifik (Seoul)	sts.ap-northeast-2.amazonaws.com	⊘Ya	⊘Ya
Asia Pasifik (Singapura)	sts.ap-southeast-1.amazonaws.com	⊘Ya	⊘Ya
Asia Pasifik (Sydney)	sts.ap-southeast-2.amazonaws.com	⊘Ya	⊘Ya
Asia Pasifik (Thailand)	sts.ap-southeast-7.amazonaws.com	⊗Tidak[1]	⊗Tidak
Asia Pasifik (Tokyo)	sts.ap-northeast-1.amazonaws.com	⊘Ya	⊘Ya
Kanada (Pusat)	sts.ca-central-1.amazonaws.com	⊘Ya	⊘Ya
Kanada Barat (Calgary)	sts.ca-west-1.amazonaws.com	⊗Tidak[1]	⊗Tidak
Tiongkok (Beijing)	sts.cn-north-1.amazonaws.com.cn	⊘Ya[2]	⊗Tidak
Tiongkok (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	⊘Ya[2]	⊘Ya
Eropa (Frankfurt)	sts.eu-central-1.amazonaws.com	⊘Ya	⊘Ya
Eropa (Irlandia)	sts.eu-west-1.amazonaws.com	⊘Ya	⊘Ya

Nama wilayah	Titik akhir	Aktif secara default	Mengaktifkan/menonaktifkan secara manual
Eropa (London)	sts.eu-west-2.amazonaws.com	✓Ya	✓Ya
Europe (Milan)	sts.eu-south-1.amazonaws.com	✗Tidak[1]	✗Tidak
Eropa (Paris)	sts.eu-west-3.amazonaws.com	✓Ya	✓Ya
Eropa (Spanyol)	sts.eu-south-2.amazonaws.com	✗Tidak[1]	✗Tidak
Eropa (Stockholm)	sts.eu-north-1.amazonaws.com	✓Ya	✓Ya
Europe (Zurich)	sts.eu-central-2.amazonaws.com	✗Tidak[1]	✗Tidak
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	✗Tidak[1]	✗Tidak
Meksiko (Tengah)	sts.mx-central-1.amazonaws.com	✗Tidak[1]	✗Tidak
Timur Tengah (Bahrain)	sts.me-south-1.amazonaws.com	✗Tidak[1]	✗Tidak
Timur Tengah (UAE)	sts.me-central-1.amazonaws.com	✗Tidak[1]	✗Tidak
Amerika Selatan (Sao Paulo)	sts.sa-east-1.amazonaws.com	✓Ya	✓Ya

[1]Anda harus mengaktifkan Wilayah untuk menggunakannya. Ini secara otomatis mengaktifkan. AWS STS Anda tidak dapat mengaktifkan atau menonaktifkan secara manual AWS STS di Wilayah ini.

[2]Untuk digunakan AWS di Tiongkok, Anda memerlukan akun dan kredensyal khusus di Tiongkok. AWS

AWS STS perubahan titik akhir global

AWS telah membuat perubahan pada AWS Security Token Service (AWS STS) titik akhir global (`https://sts.amazonaws.com`) di Wilayah yang diaktifkan secara default untuk meningkatkan ketahanan dan kinerjanya. Sebelumnya, semua permintaan ke titik akhir AWS STS global dilayani oleh satu Wilayah AWS, US East (Virginia N.). Sekarang di Wilayah diaktifkan secara default, permintaan ke titik akhir AWS STS global secara otomatis disajikan di Wilayah yang sama tempat permintaan berasal, bukan Wilayah AS Timur (Virginia Utara). Perubahan ini tidak akan diterapkan ke Wilayah keikutsertaan.

Dengan perubahan ini, AWS STS akan memproses permintaan Anda berdasarkan Region asal dan DNS resolver yang digunakan. Permintaan ke titik akhir AWS STS global disajikan di Wilayah yang sama dengan beban kerja AWS yang Anda gunakan jika permintaan DNS untuk titik akhir AWS STS global ditangani oleh server DNS Amazon di Wilayah yang diaktifkan secara default. Permintaan ke titik akhir AWS STS global akan terus dilayani di Wilayah AS Timur (Virginia Utara) jika permintaan Anda berasal dari Wilayah keikutsertaan atau jika permintaan Anda diselesaikan menggunakan penyelesai DNS selain server DNS Amazon. Untuk informasi selengkapnya tentang Amazon DNS, lihat Server DNS Amazon di Panduan Pengguna Amazon Virtual Private Cloud.

Tabel berikut menunjukkan bagaimana permintaan ke titik akhir AWS STS global dirutekan berdasarkan penyedia DNS Anda.

DNS Resolver	Permintaan ke titik akhir AWS STS global diarahkan ke lokal? Wilayah AWS
Amazon DNS resolver di Amazon VPC di Wilayah diaktifkan secara default	Ya
Penyelesai DNS Amazon di VPC Amazon di Wilayah keikutsertaan	Tidak, permintaan akan dialihkan ke Wilayah AS Timur (Virginia N.)
DNS resolver yang disediakan oleh ISP Anda, penyedia DNS publik, atau penyedia DNS lainnya	Tidak, permintaan akan dialihkan ke Wilayah AS Timur (Virginia N.)

Untuk memastikan gangguan minimal terhadap proses yang ada, AWS telah menerapkan langkah-langkah berikut:

- AWS CloudTrail log untuk permintaan yang dibuat ke titik akhir AWS STS global dikirim ke Wilayah AS Timur (Virginia N.). CloudTrail log untuk permintaan yang dilayani oleh titik akhir AWS STS Regional akan terus dicatat ke Wilayah masing-masing di CloudTrail.
- CloudTrail log untuk operasi yang dilakukan oleh titik akhir AWS STS global dan titik akhir Regional memiliki bidang tambahan `endpointType` dan `awsServingRegion` untuk menunjukkan titik akhir dan Wilayah mana yang melayani permintaan. Untuk contoh CloudTrail log, lihatContoh AWS STS Acara API menggunakan titik akhir global dalam file CloudTrail log.
- Permintaan yang dibuat ke titik akhir AWS STS global memiliki nilai `us-east-1` untuk kunci `aws:RequestedRegion` kondisi, terlepas dari Wilayah mana yang melayani permintaan tersebut.
- Permintaan yang ditangani oleh titik akhir AWS STS global tidak membagikan kuota permintaan per detik dengan titik akhir Regional. AWS STS

Jika Anda memiliki beban kerja di Wilayah keikutsertaan dan masih menggunakan titik akhir AWS STS global, kami sarankan untuk bermigrasi ke titik akhir AWS STS regional untuk meningkatkan ketahanan dan kinerja. Untuk informasi selengkapnya tentang mengonfigurasi AWS STS titik akhir regional, lihat Titik akhir AWS STS Regional di Panduan Referensi Alat AWS SDKs dan Alat.

AWS CloudTrail dan titik akhir Regional

Panggilan ke titik akhir regional dan global dicatat di `tlsDetails` bidang di AWS CloudTrail. Panggilan ke titik akhir regional, seperti`us-east-2.amazonaws.com`, masuk CloudTrail ke wilayah yang sesuai. Memanggil ke titik akhir global, `sts.amazonaws.com`, dicatat sebagai panggilan ke layanan global. Peristiwa untuk AWS STS titik akhir global dicatat ke us-east-1.

> **Note**
>
> `tlsDetails`hanya dapat dilihat untuk layanan yang mendukung bidang ini. Lihat Layanan yang mendukung detail TLS CloudTrail di AWS CloudTrail Panduan Pengguna
> Lihat informasi yang lebih lengkap di Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail.

Aktifkan akses broker identitas khusus ke AWS konsol

Anda dapat menulis dan menjalankan kode untuk membuat URL yang memungkinkan pengguna yang masuk ke jaringan organisasi Anda mengakses Konsol Manajemen AWS. URL menyertakan

token masuk yang Anda dapatkan dari AWS dan yang mengotentikasi pengguna. AWS Sesi konsol yang dihasilkan mungkin termasuk yang berbeda `AccessKeyId` karena federasi. Untuk melacak penggunaan kunci akses untuk masuk federasi melalui CloudTrail acara terkait, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail dan acara Konsol Manajemen AWS masuk.

> **Note**
>
> Jika organisasi Anda menggunakan penyedia identitas (IdP) yang kompatibel dengan SAML, Anda dapat menyiapkan akses ke konsol tanpa menulis kode. Ini berfungsi dengan penyedia seperti Layanan Federasi Direktori Aktif Microsoft atau Shibboleth sumber terbuka. Untuk detailnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS.

Untuk memungkinkan pengguna organisasi Anda mengakses Konsol Manajemen AWS, Anda dapat membuat broker identitas khusus yang melakukan langkah-langkah berikut:

1. Pastikan bahwa pengguna diautentikasi oleh sistem identitas lokal Anda.
2. Panggil operasi AWS Security Token Service (AWS STS) AssumeRole (disarankan) atau GetFederationToken API untuk mendapatkan kredenSIAL keamanan sementara bagi pengguna. Untuk mempelajari tentang berbagai metode yang dapat Anda gunakan untuk menjalankan peran, lihat Metode untuk mengambil peran. Untuk mempelajari cara mengirimkan tag sesi opsional saat Anda mendapatkan kredensial keamanan Anda, lihat Lewati tag sesi di AWS STS.
   - Jika Anda menggunakan salah satu dari Operasi API `AssumeRole*` untuk mendapatkan kredensial keamanan sementara untuk sebuah peran, Anda dapat menyertakan parameter `DurationSeconds` dalam panggilan Anda. Parameter ini menentukan durasi sesi peran Anda, dari 900 detik (15 menit) hingga setelan durasi sesi maksimum untuk peran tersebut. Saat Anda menggunakan `DurationSeconds` dalam `AssumeRole*` operasi, Anda harus memanggilnya sebagai pengguna IAM dengan kredenSIAL jangka panjang. Jika tidak, panggilan ke titik akhir federasi pada langkah 3 gagal. Untuk mempelajari cara melihat atau mengubah nilai maksimum untuk peran, lihat Memperbarui durasi sesi maksimum untuk peran.
   - Jika Anda menggunakan operasi API `GetFederationToken` untuk mendapatkan kredensial, Anda bisa menyertakan parameter `DurationSeconds` dalam panggilan Anda. Parameter ini menentukan durasi sesi peran Anda. Nilai dapat berkisar dari 900 detik (15 menit) hingga 129.600 detik (36 jam). Anda dapat melakukan panggilan API ini hanya dengan menggunakan kredenSIAL AWS keamanan jangka panjang dari pengguna IAM. Anda juga dapat melakukan panggilan ini menggunakan Pengguna root akun AWS kredenSIAL, tetapi kami tidak

merekomendasikannya. Jika Anda membuat panggilan ini sebagai pengguna akar, sesi default berlangsung selama satu jam. Atau Anda dapat menentukan sesi dari 900 detik (15 menit) hingga 3.600 detik (satu jam).

3. Panggil titik akhir AWS federasi dan berikan kredenSIAL keamanan sementara untuk meminta token masuk.
4. Buat URL untuk konsol yang mencakup token:
   - Jika Anda menggunakan salah satu dari operasi API `AssumeRole*` di URL Anda, Anda dapat menyertakan Parameter HTTP `SessionDuration`. Parameter ini menentukan durasi sesi konsol, dari 900 detik (15 menit) hingga 43200 detik (12 jam).
   - Jika Anda menggunakan operasi API `GetFederationToken` di URL Anda, Anda dapat menyertakan Parameter `DurationSeconds`. Parameter ini menentukan durasi sesi konsol federasi. Nilai dapat berkisar dari 900 detik (15 menit) hingga 129.600 detik (36 jam).

   > **Note**
   >
   > - Anda `SessionDuration` tidak boleh lebih besar dari atau sama dengan pengaturan durasi sesi maksimum untuk peran yang Anda ambil. Misalnya, Anda telah menetapkan durasi sesi maksimum untuk peran yang ingin Anda asumsikan menjadi 5 jam. `SessionDuration`Parameter Anda bisa 16524 detik atau 4 jam 59 detik.
   > - Jangan gunakan parameter `SessionDuration` HTTP saat Anda mendapatkan kredenSIAL sementara dengan`GetFederationToken`. Operasi akan gagal.
   > - Menggunakan kredenSIAL untuk satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Saat Anda menggunakan peran untuk memberikan izin ke aplikasi yang berjalan pada instans EC2, aplikasi tersebut tidak terkena batasan ini.
   > - Jangan gunakan parameter `SessionDuration` HTTP saat Anda mendapatkan kredenSIAL sementara melalui rantai peran. Operasi akan gagal.

5. Berikan URL ke pengguna atau panggil URL atas nama pengguna.

URL yang disediakan oleh titik akhir federasi berlaku selama 15 menit setelah dibuat. Ini berbeda dari durasi (dalam detik) sesi kredensial keamanan sementara yang dikaitkan dengan URL. Kredensial tersebut berlaku selama durasi yang Anda tentukan saat membuatnya, dimulai sejak saat informasi tersebut dibuat.

> **Important**
>
> URL memberikan akses ke AWS sumber daya Anda melalui Konsol Manajemen AWS jika Anda telah mengaktifkan izin di kredentif keamanan sementara terkait. Karena alasan ini, Anda harus memperlakukan URL sebagai rahasia. Kami menyarankan mengembalikan URL melalui pengalihan aman, misalnya, dengan menggunakan kode status respons HTTP 302 melalui koneksi SSL. Untuk informasi selengkapnya tentang kode status respons HTTP 302, lihat RFC 2616, bagian 10.3.3.

Untuk menyelesaikan tugas ini, Anda dapat menggunakan HTTPS Query API untuk AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS). Atau, Anda dapat menggunakan bahasa pemrograman, seperti Java, Ruby, atau C#, bersama dengan SDK AWS yang sesuai. Masing-masing metode ini dijelaskan dalam topik berikut.

Topik

- Contoh kode menggunakan operasi API kueri IAM
- Contoh kode menggunakan Python
- Contoh kode menggunakan Java
- Contoh menunjukkan cara membangun URL (Ruby)

Contoh kode menggunakan operasi API kueri IAM

Anda dapat membuat URL yang memberikan peran dan prinsipal federasi akses langsung ke. Konsol Manajemen AWS Tugas ini menggunakan IAM dan AWS STS HTTPS Query API. Untuk informasi selengkapnya tentang membuat permintaan pertanyaan, lihat Membuat Permintaan Kueri.

> **Note**
>
> Prosedur berikut berisi contoh string teks. Untuk meningkatkan keterbacaan, jeda baris telah ditambahkan ke beberapa contoh yang lebih panjang. Saat Anda membuat string ini untuk Anda gunakan sendiri, Anda harus mengurangi setiap pecahan baris.

Untuk memberikan peran dan prinsipal federasi akses ke sumber daya Anda dari Konsol Manajemen AWS

1. Autentikasi pengguna di sistem identitas dan otorisasi Anda.
2. Dapatkan kredensial keamanan sementara untuk pengguna. KredenSIAL sementara terdiri dari ID kunci akses, kunci akses rahasia, dan token sesi. Untuk informasi lebih lanjut tentang membuat kredensial sementara, lihat Kredensial keamanan sementara di IAM.

   Untuk mendapatkan kredenSIAL sementara, Anda memang AWS STS AssumeRole gil API (disarankan) atau GetFederationToken API. Untuk informasi selengkapnya tentang perbedaan antara operasi API ini, lihat Mem ahami Opsi API untuk Mendelegasikan Akses dengan Aman ke AWS Akun Anda di Blog AWS Keamanan.

   > **Important**
   >
   > Saat Anda menggunakan API GetFederationToken untuk membuat kredensial keamanan sementara, Anda harus menentukan izin yang diberikan kredensial tersebut kepada pengguna yang memegang peran tersebut. Untuk setiap operasi API yang dimulai dengan `AssumeRole*`, Anda menggunakan peran IAM untuk menetapkan izin. Untuk operasi API lainnya, mekanisme bervariasi dengan API. Untuk lebih detailnya, lihat Izin untuk kredensial keamanan sementara. Sebagai tambahan, jika Anda menggunakan operasi API `AssumeRole*`, Anda harus menyebutnya sebagai pengguna IAM dengan kredensial jangka panjang. Jika tidak, panggilan ke titik akhir federasi pada langkah 3 gagal.
3. Setelah Anda mendapatkan kredensial keamanan sementara, masukkan ke dalam rangkaian sesi JSON untuk menukarnya dengan token masuk. Contoh berikut menunjukkan cara mengkode kredensial. Anda mengganti teks placeholder dengan nilai yang sesuai dari kredensial yang Anda terima di langkah sebelumnya.

   ```
   {"sessionId":"*** temporary access key ID ***",
   "sessionKey":"*** temporary secret access key ***",
   "sessionToken":"*** session token ***"}
   ```
4. URL dikode string sesi dari langkah sebelumnya. Karena informasi yang Anda encoding itu sensitif, kami menganjurkan agar Anda menghindari penggunaan layanan web untuk encoding ini. Alih-alih, gunakan fungsi atau fitur yang dipasang secara lokal di perangkat pengembangan Anda untuk mengenkode informasi ini dengan aman. Anda dapat menggunakan

fungsi `urllib.quote_plus` di Python, fungsi `URLEncoder.encode` di Java, atau fungsi `CGI.escape` di Ruby. Lihat contoh-contoh ini nanti dalam topik ini.

5.

> **Note**
>
> AWS mendukung permintaan POST di sini.

Kirim permintaan Anda ke titik akhir AWS federasi:

`https://region-code.signin.aws.amazon.com/federation`

Untuk daftar *region-code* nilai yang mungkin, lihat kolom Wilayah di titik akhir AWS Sign-In titik akhir Anda dapat secara opsional menggunakan titik akhir AWS Sign-In federasi default:

`https://signin.aws.amazon.com/federation`

Permintaan tersebut harus menyertakan parameter `Action` dan `Session`, dan (secara opsional) jika Anda menggunakan operasi API `AssumeRole*`, sebuah parameter HTTP `SessionDuration` seperti ditunjukkan dalam contoh berikut.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Petunjuk berikut dalam langkah ini hanya berfungsi menggunakan permintaan GET.

Parameter HTTP `SessionDuration` ini menentukan durasi sesi konsol federasi. Ini terpisah dari durasi kredensial sementara yang Anda tentukan menggunakan parameter `DurationSeconds`. Anda dapat menentukan nilai maksimal `SessionDuration` adalah 43.200 (12 jam). Jika `SessionDuration` parameter hilang, maka sesi default ke durasi kredenSIAL yang Anda ambil AWS STS pada langkah 2 (yang defaultnya adalah satu jam). Lihat dokumentasi untuk `AssumeRole` API untuk detail tentang cara menentukan durasi menggunakan parameter `DurationSeconds`. Kemampuan untuk membuat sesi konsol yang lebih lama dari satu jam bersifat intrinsik bagi `getSigninToken` operasi titik akhir federasi.

> **Note**
>
> - Anda `SessionDuration` tidak boleh lebih besar dari atau sama dengan pengaturan durasi sesi maksimum untuk peran yang Anda ambil. Misalnya, Anda telah menetapkan durasi sesi maksimum untuk peran yang ingin Anda asumsikan menjadi 5 jam. `SessionDuration`Parameter Anda bisa 16524 detik atau 4 jam 59 detik.
> - Jangan gunakan parameter `SessionDuration` HTTP saat Anda mendapatkan kredenSIAL sementara dengan`GetFederationToken`. Operasi akan gagal.
> - Menggunakan kredenSIAL untuk satu peran untuk mengambil peran yang berbeda disebut rantai peran. Saat Anda menggunakan rantai peran, kredensial baru Anda dibatasi hingga durasi maksimum satu jam. Saat Anda menggunakan peran untuk memberikan izin ke aplikasi yang berjalan pada instans EC2, aplikasi tersebut tidak terkena batasan ini.
> - Jangan gunakan parameter `SessionDuration` HTTP saat Anda mendapatkan kredenSIAL sementara melalui rantai peran. Operasi akan gagal.

Saat Anda mengaktifkan sesi konsol dengan durasi yang diperpanjang, Anda meningkatkan risiko eksposur kredensial. Untuk membantu Anda mengurangi risiko ini, Anda dapat segera menonaktifkan sesi konsol aktif untuk peran apa pun dengan memilih Mencabut Sesi pada Ringkasan Peran IAM halaman konsol. Untuk informasi selengkapnya, lihat Mencabut kredensi keamanan sementara peran IAM.

Berikut ini adalah contoh seperti apa permintaan Anda. Garis dibungkus di sini agar mudah dibaca, tetapi Anda harus mengirimkannya sebagai string satu baris.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

Respons dari titik akhir federasi adalah dokumen JSON dengan nilai `SigninToken`. Ini akan terlihat serupa dengan contoh berikut.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS mendukung permintaan POST di sini.

Terakhir, buat URL yang dapat digunakan pengguna Anda untuk mengakses Konsol Manajemen AWS. URL adalah titik akhir URL federasi yang sama yang Anda gunakan di Step 5, ditambah parameter berikut:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Petunjuk berikut dalam langkah ini hanya berfungsi menggunakan GET API.

Contoh berikut menunjukkan seperti apa URL akhir itu. URL valid selama 15 menit sejak dibuat. Kredensial keamanan sementara dan sesi konsol yang disematkan dalam URL valid selama durasi yang Anda tentukan di parameter HTTP `SessionDuration` saat Anda memintanya untuk pertama kali.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
```

```
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Contoh kode menggunakan Python

Contoh berikut menunjukkan cara menggunakan Python untuk secara terprogram membangun URL yang memberi pengguna akses langsung ke. Konsol Manajemen AWS Berikut ini adalah dua contoh:

- Gabungkan melalui permintaan GET ke AWS
- Federasi melalui permintaan POST ke AWS

Kedua contoh menggunakan AssumeRole API AWS SDK untuk Python (Boto3) dan untuk mendapatkan kredenSIAL keamanan sementara.

Jangan sertakan `SessionDuration` jika `AssumeRoleSession` kredensi Anda berasal dari rantai peran. Jika Anda menyertakan`SessionDuration`, operasi akan gagal.

Gunakan Permintaan GET

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Akun AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')
```

```
assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
```

```
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Gunakan Permintaan POST

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your A Akun AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
```

```
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
```

```
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest)
input("Press Enter to close the browser window...")
```

Contoh kode menggunakan Java

Contoh berikut menunjukkan bagaimana menggunakan Java untuk secara terprogram membangun URL yang memberi pengguna akses langsung ke. Konsol Manajemen AWS Ptongan kode berikut menggunakan SDK for Java AWS.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
```

```
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Action\":\"sns:*
\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
```

```
                            "?Action=getSigninToken" +
                            "&DurationSeconds=43200" +
                            "&SessionType=json&Session=" +
                            URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Contoh menunjukkan cara membangun URL (Ruby)

Contoh berikut menunjukkan bagaimana menggunakan Ruby untuk secara terprogram membangun URL yang memberi pengguna akses langsung ke. Konsol Manajemen AWS Potongan kode ini menggunakan SDK for Ruby AWS.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
```

```
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",       \"Statement\":{\"Effect\":\"Allow\",
\"Action\":\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
```

```
                        "&SessionType=json&Session=" +
                        CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Tag untuk AWS Identity and Access Management sumber daya

Tanda adalah label atribut khusus yang dapat Anda tetapkan ke sumber daya AWS . Setiap tag memiliki dua bagian:

- Sebuah kunci tag (misalnya, `CostCenter`, `Environment`, `Project`, atau `Purpose`).
- Bidang opsional yang dikenal sebagai nilai tag (misalnya, `111122223333`, `Production`, atau nama tim). Mengabaikan nilai tag sama dengan menggunakan rangkaian kosong.

Bersama-sama ini dikenal sebagai pasangan nilai-kunci. Untuk batasan jumlah tag yang dapat Anda miliki pada sumber daya IAM, lihatIAM dan AWS STS kuota.

> **Note**
>
> Untuk detail tentang sensitivitas huruf besar/kecil untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Tag membantu Anda mengidentifikasi dan mengatur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait. Misalnya, Anda

dapat menetapkan tag yang sama ke peran IAM yang Anda tetapkan ke bucket Amazon S3. Untuk informasi selengkapnya tentang strategi penandaan, lihat Panduan Pengguna AWS sumber daya penandaan.

Selain mengidentifikasi, mengatur, dan melacak sumber daya IAM Anda dengan tag, Anda dapat menggunakan tag dalam kebijakan IAM untuk membantu mengontrol siapa yang dapat melihat dan berinteraksi dengan sumber daya Anda. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Topik

- Pilih AWS konvensi penamaan tag
- Aturan untuk menandai di IAM dan AWS STS
- Menandai pengguna IAM
- Kata kunci IAM rolls
- Tandai kebijakan yang dikelola pelanggan
- Menandai penyedia identitas OpenID Connect (OIDC)
- Kata kunci IAM SAML identity provider
- Tandai profil instans untuk peran Amazon EC2
- Sertifikat server tag
- Menandai perangkat MFA virtual
- Tandai instans manajer akses akun
- Lewati tag sesi di AWS STS

Pilih AWS konvensi penamaan tag

Saat Anda mulai melampirkan tag ke sumber daya IAM Anda, pilih konvensi penamaan tag Anda dengan hati-hati. Terapkan konvensi yang sama untuk semua AWS tag Anda. Ini sangat penting jika Anda menggunakan tag dalam kebijakan untuk mengontrol akses ke AWS sumber daya. Jika Anda sudah menggunakan tag di AWS, tinjau konvensi penamaan Anda dan sesuaikan sesuai kebutuhan.

> **Note**
>
> Jika akun Anda adalah anggota AWS Organizations, lihat Kebijakan tag di panduan AWS Organizations pengguna untuk mempelajari lebih lanjut tentang menggunakan tag di AWS Organizations.

Praktik terbaik untuk penamaan tag

Ini adalah beberapa praktik terbaik dan konvensi penamaan untuk tag.

Pastikan nama tag digunakan secara konsisten. Misalnya, tag `CostCenter` dan `costcenter` berbeda, sehingga satu mungkin dikonfigurasi sebagai tag alokasi biaya untuk analisis dan pelaporan keuangan dan yang lainnya mungkin tidak. Demikian pula, `Name` tag muncul di AWS Konsol untuk banyak sumber daya, tetapi `name` tag tidak. Untuk detail tentang sensitivitas huruf besar/kecil untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Sejumlah tag ditentukan sebelumnya oleh AWS atau dibuat secara otomatis oleh berbagai AWS layanan. Banyak AWS nama tag yang ditentukan menggunakan semua huruf kecil, dengan tanda hubung yang memisahkan kata-kata dalam nama, dan awalan untuk mengidentifikasi layanan sumber untuk tag. Contoh:

- `aws:ec2spot:fleet-request-id`mengidentifikasi Permintaan Instans Spot Amazon EC2 yang meluncurkan instans.
- `aws:cloudformation:stack-name`mengidentifikasi tum CloudFormation pukan yang membuat sumber daya.
- `elasticbeanstalk:environment-name`mengidentifikasi aplikasi yang membuat sumber daya.

Pertimbangkan untuk menamai tag Anda menggunakan semua huruf kecil, dengan tanda hubung yang memisahkan kata, dan awalan yang mengidentifikasi nama organisasi atau nama yang disingkat. Misalnya, untuk perusahaan fiktif bernama AnyCompany, Anda dapat menentukan tag seperti:

- `anycompany:cost-center`untuk mengidentifikasi kode Pusat Biaya internal
- `anycompany:environment-type`untuk mengidentifikasi apakah lingkungan adalah pengembangan, pengujian, atau produksi
- `anycompany:application-id`untuk mengidentifikasi aplikasi sumber daya diciptakan

Awalan memastikan bahwa tag diidentifikasi dengan jelas sebagai telah ditentukan oleh organisasi Anda dan bukan oleh AWS atau alat pihak ketiga yang mungkin Anda gunakan. Menggunakan semua huruf kecil dengan tanda hubung untuk pemisah menghindari kebingungan tentang cara menggunakan huruf besar nama tag. Misalnya, `anycompany:project-id` lebih mudah diingat daripada`ANYCOMPANY:ProjectID`,`anycompany:projectID`, atau`Anycompany:ProjectId`.

Aturan untuk menandai di IAM dan AWS STS

Sejumlah konvensi mengatur pembuatan dan penerapan tag di IAM dan AWS STS.

Penamaan tag

Perhatikan konvensi berikut saat merumuskan konvensi penamaan tag untuk sumber daya IAM, sesi AWS STS peran asumsi, dan AWS STS sesi pengguna federasi:

Persyaratan karakter — Kunci dan nilai tag dapat mencakup kombinasi huruf, angka, spasi, dan simbol _.:/= + - @.

Sensitivitas huruf besar/kecil — Sensitivitas huruf besar/kecil untuk kunci tag berbeda tergantung pada jenis sumber daya IAM yang diberi tag. Nilai kunci tag untuk pengguna dan peran IAM tidak peka huruf besar/kecil, tetapi huruf besar dipertahankan. Ini berarti Anda tidak dapat memisahkan kunci tag **`Department`** dan **`department`**. Jika Anda sudah memberi tag ke pengguna dengan tag **`Department=finance`** dan Anda menambahkan tag **`department=hr`**, itu menggantikan tag pertama. Tag kedua tidak ditambahkan.

Untuk jenis sumber daya IAM lainnya, nilai kunci tag peka huruf besar/kecil. Ini berarti Anda dapat memisahkan kunci tanda **`Costcenter`** dan **`costcenter`**. Sebagai contoh, jika Anda telah menandai kebijakan yang dikelola pelanggan dengan tanda **`Costcenter = 1234`** dan Anda menambahkan tanda **`costcenter = 5678`**, kebijakan akan memiliki kunci tanda **`Costcenter`** dan **`costcenter`**.

Sebagai praktik terbaik, kami menyarankan agar Anda menghindari penggunaan tanda yang serupa dengan perlakuan huruf besar kecil yang tidak konsisten. Sebaiknya Anda memutuskan strategi untuk menggunakan huruf kapital tag, dan menerapkan strategi tersebut secara konsisten di semua jenis sumber daya. Untuk mempelajari lebih lanjut tentang praktik terbaik untuk penandaan, lihat Men andai Sumber AWS Daya di. Referensi Umum AWS

Daftar berikut menunjukkan perbedaan sensitivitas huruf besar/kecil untuk kunci tag yang dilampirkan ke sumber daya IAM.

Nilai kunci tanda tidak peka huruf besar kecil:

- Peran IAM
- Pengguna IAM:

Nilai kunci tanda peka huruf besar kecil:

- Kebijakan yang dikelola pelanggan
- Profil instans
- Penyedia identitas OpenID Connect
- Penyedia identitas SAML
- Sertifikat server
- Perangkat MFA virtual

Selain itu, aturan berikut ini berlaku:

- Anda tidak dapat membuat kunci tanda atau nilai yang dimulai dengan teks **`aws:`**. Awalan tag ini dicadangkan untuk penggunaan AWS internal.
- Anda dapat membuat tanda dengan nilai kosong seperti**`phoneNumber =`** . Anda tidak dapat membuat kunci tanda kosong.
- Anda tidak dapat menentukan beberapa nilai dalam satu tag, tetapi Anda dapat membuat struktur multinilai khusus dalam satu nilai. Misalnya, anggap bahwa pengguna Zhang bekerja di tim engineer dan tim QA. Jika Anda melampirkan atnda **`team = Engineering`** lalu melampirkan tanda **`team = QA`**, Anda mengubah nilai dari tanda dari **`Engineering`** ke **`QA`**. Sebaliknya, Anda dapat menyertakan beberapa nilai dalam satu tanda dengan pemisah kustom. Dalam contoh ini, Anda dapat memberikan tag **`team = Engineering:QA`** ke Zhang.

> **Note**
>
> Untuk mengontrol akses ke engineer dalam contoh ini menggunakan tanda **`team`**, Anda harus membuat kebijakan yang memungkinkan setiap konfigurasi yang mungkin mencakup **`Engineering`**, termasuk **`Engineering:QA`**. Untuk mem-pelajari selengkapnya tentang penggunaan tanda dalam kebijakan, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Menerapkan dan mengubah tag

Perhatikan konvensi berikut saat melampirkan tag ke sumber daya IAM:

- Anda dapat menandai sebagian besar sumber daya IAM, tetapi tidak grup, peran yang diasumsikan, laporan akses, atau perangkat MFA berbasis perangkat keras.
- Anda tidak dapat menggunakan Editor Tag untuk menandai sumber daya IAM. Editor Tag tidak mendukung tag IAM. Untuk informasi tentang penggunaan Editor Tanda dengan layanan lain, lihat Bekerja dengan Editor Tanda dalam Panduan Pengguna AWS Resource Groups .
- Untuk menandai sumber daya IAM, Anda harus memiliki izin tertentu. Untuk memberi atau menghapus tanda sumber daya, Anda juga harus memiliki izin untuk mencantumkan tanda. Untuk informasi selengkapnya, lihat daftar topik untuk setiap sumber daya IAM di akhir halaman ini.
- Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.
- Anda dapat menerapkan tag yang sama ke beberapa sumber daya IAM. Misalnya, Anda memiliki departemen bernama `AWS_Development` dengan 12 anggota. Anda bisa memiliki 12 pengguna dan satu peran dengan kunci tag **`department`** dan nilai **`awsDevelopment`** (**`department = awsDevelopment`**). Anda juga dapat menggunakan tag yang sama pada sumber daya di layanan yang mendukung pemberian tag.
- Entitas IAM (pengguna atau peran) tidak dapat memiliki beberapa instance kunci tag yang sama. Misalnya, jika Anda memiliki pengguna dengan tag key-value pair**`costCenter = 1234`**, Anda kemudian dapat melampirkan tag key-value pair. **`costCenter = 5678`** IAM memperbarui nilai **`costCenter`** tag ke**`5678`**.
- Untuk mengedit tag yang dilampirkan ke entitas IAM (pengguna atau peran), lampirkan tag dengan nilai baru untuk mengganti tag yang ada. Sebagai contoh, anggap Anda memiliki pengguna dengan pasangan nilai-kunci tanda **`department = Engineering`**. Jika perlu memindahkan pengguna ke departemen QA, Anda dapat melampirkan pasangan nilai-kunci tanda **`department = QA`** ke pengguna. Ini menghasilkan nilai **`Engineering`** dari kunci tag **`department`** yang digantikan dengan nilai **`QA`**.

Menandai pengguna IAM

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke pengguna IAM. Misalnya, untuk menambahkan informasi lokasi ke pengguna, Anda dapat menambahkan kunci tag **`location`** dan nilai tanda **`us_wa_seattle`**. Atau Anda bisa menggunakan tiga pasangan kunci nilai tanda lokasi yang terpisah: **`loc-country = us`**, **`loc-state = wa`**, dan

`loc-city = seattle`. Anda dapat menggunakan tanda untuk mengontrol akses pengguna ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke pengguna. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin diperlukan untuk menandai pengguna IAM

Anda harus mengonfigurasi izin untuk mengizinkan pengguna IAM menandai pengguna lain. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Untuk mengizinkan pengguna IAM menambah, membuat daftar, atau menghapus tag untuk pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk pengguna IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk mengizinkan pengguna IAM untuk mengelola tag sendiri

Tambahkan pernyataan berikut ini ke kebijakan izin untuk pengguna agar pengguna dapat mengelola tag mereka sendiri. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh

dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Untuk mengizinkan pengguna IAM menambahkan tag ke pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk pengguna IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk pengguna tertentu.

> **Note**
>
> Tindakan `iam:TagUser` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListUserTags`

Untuk menggunakan kebijakan ini, ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada pengguna IAM (konsol)

Anda dapat mengelola tag untuk pengguna IAM dari Konsol Manajemen AWS.

Untuk mengelola tanda pada pengguna (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran lalu pilih nama pengguna yang ingin Anda edit.
3. Pilih Tag kemudian selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag baru jika pengguna belum memiliki tag.
   - Pilih Kelola tag untuk mengelola kumpulan tag yang ada.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada pengguna IAM (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk pengguna IAM. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag bagi pengguna IAM.

Untuk mencantumkan tag yang saat ini dilampirkan ke pengguna IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam list-user-tags
- AWS API: ListUserTags

Untuk melampirkan tag ke pengguna IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-user
- AWS API: TagUser

Untuk menghapus tag dari pengguna IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-user
- AWS API: UntagUser

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Kata kunci IAM rolls

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke peran IAM. Misalnya, untuk menambahkan informasi lokasi ke peran, Anda dapat menambahkan kunci tag **`location`** dan nilai tanda **`us_wa_seattle`**. Atau Anda bisa menggunakan tiga pasangan kunci nilai tanda lokasi yang terpisah: **`loc-country = us`**, **`loc-state = wa`**, dan **`loc-city = seattle`**. Anda dapat menggunakan tanda untuk mengontrol akses peran ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke peran. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin diperlukan untuk menandai peran IAM

Anda harus mengonfigurasi izin untuk mengizinkan peran IAM untuk menandai entitas lain (pengguna atau peran). Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Untuk mengizinkan peran IAM menambah, mencantumkan, atau menghapus tag untuk pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk peran IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<username>`* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk mengizinkan peran IAM untuk menambahkan tag ke pengguna tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk peran IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk pengguna tertentu.

Untuk menggunakan kebijakan ini, ganti *<username>* dengan nama pengguna yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Untuk mengizinkan peran IAM menambah, mencantumkan, atau menghapus tag untuk peran tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk peran IAM yang perlu mengelola tag. Ganti *<rolename>* dengan nama peran yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
```

```
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada peran IAM (konsol)

Anda dapat mengelola tag untuk peran IAM dari Konsol Manajemen AWS.

Untuk mengelola tanda pada peran (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Peran lalu pilih nama peran yang ingin Anda edit.
3. Pilih Tag kemudian selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag baru jika peran tersebut belum memiliki tag.
   - Pilih Kelola tag untuk mengelola kumpulan tag yang ada.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Kemudian, pilih S impan perubahan.

Mengelola tag pada peran IAM (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk peran IAM. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk peran IAM.

Untuk mencantumkan tag yang saat ini dilampirkan ke peran IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam list-role-tags
- AWS API: ListRoleTags

Untuk melampirkan tag ke peran IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-role
- AWS API: TagRole

Untuk menghapus tag dari peran IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-role
- AWS API: UntagRole

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Tandai kebijakan yang dikelola pelanggan

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke kebijakan yang dikelola pelanggan Anda. Misalnya, untuk menandai kebijakan dengan informasi departemen, Anda dapat menambahkan kunci tanda **`Department`** dan nilai tanda **`eng`**. Atau, Anda mungkin ingin menandai kebijakan guna menunjukkan bahwa kebijakan tersebut adalah untuk lingkungan tertentu, seperti **`Environment = lab`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke identitas. Untuk mempelajari selengkapnya tentang penggunaan tanda untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin yang diperlukan untuk menandai kebijakan terkelola pelanggan

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) menandai kebijakan yang dikelola pelanggan. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListPolicyTags`
- `iam:TagPolicy`
- `iam:UntagPolicy`

Untuk mengizinkan entitas IAM (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk kebijakan yang dikelola pelanggan

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<policyname>* dengan nama kebijakan yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke kebijakan yang dikelola pelanggan tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk kebijakan tertentu.

> **Note**
>
> Tindakan `iam:TagPolicy` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListPolicyTags`

Untuk menggunakan kebijakan ini, ganti *<policyname>* dengan nama kebijakan yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada kebijakan yang dikelola pelanggan IAM (konsol)

Anda dapat mengelola tag untuk kebijakan yang dikelola pelanggan IAM dari Konsol Manajemen AWS.

Untuk mengelola tanda pada kebijakan terkelola pelanggan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Kebijakan lalu pilih nama kebijakan terkelola pelanggan yang ingin Anda edit.
3. Pilih tab Tag dan kemudian pilih Kel ola tag.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada kebijakan yang dikelola pelanggan IAM (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk kebijakan yang dikelola pelanggan IAM. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag kebijakan yang dikelola pelanggan IAM.

Untuk mencantumkan tag yang saat ini dilampirkan ke kebijakan yang dikelola pelanggan IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam list-policy-tags
- AWS API: ListPolicyTags

Untuk melampirkan tag ke kebijakan yang dikelola pelanggan IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-policy
- AWS API: TagPolicy

Untuk menghapus tag dari kebijakan yang dikelola pelanggan IAM (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-policy

- AWS API: UntagPolicy

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Menandai penyedia identitas OpenID Connect (OIDC)

Anda dapat menggunakan nilai kunci tag IAM untuk menambahkan atribut khusus ke penyedia identitas IAM OpenID Connect (OIDC). Misalnya, untuk mengidentifikasi penyedia identitas OIDC, Anda dapat menambahkan kunci tanda **`google`** dan nilai tanda **`oidc`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Izin diperlukan untuk menandai penyedia identitas IAM OIDC

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai penyedia identitas IAM OIDC. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Untuk mengizinkan entitas IAM menambah, mencantumkan, atau menghapus tag untuk penyedia identitas IAM OIDC

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<OIDCProviderName>`* dengan nama penyedia OIDC yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke penyedia identitas IAM OIDC tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk penyedia identitas tertentu.

> **Note**
>
> Tindakan `iam:TagOpenIDConnectProvider` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListOpenIDConnectProviderTags`

Untuk menggunakan kebijakan ini, ganti *<OIDCProviderName>* dengan nama penyedia OIDC yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada penyedia identitas IAM OIDC (konsol)

Anda dapat mengelola tag untuk penyedia identitas IAM OIDC dari. Konsol Manajemen AWS

Untuk mengelola tanda pada penyedia identitas OIDC (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Penyedia identitas, kemudian pilih nama penyedia identitas yang ingin Anda edit.
3. Pilih tab Tag, lalu di bagian Tag, pilih Kelola tag, lalu selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tanda jika penyedia identitas OIDC belum memiliki tanda atau untuk menambahkan tanda baru.
   - Mengedit kunci dan nilai tanda yang ada.
   - Untuk menghapus tanda, pilih Hapus tanda.
4. Lalu pilih Simpan Perubahan.

Mengelola tag pada penyedia identitas IAM OIDC (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk penyedia identitas IAM OIDC. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk penyedia identitas IAM OIDC.

Untuk mencantumkan tag yang saat ini dilampirkan ke penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: aws iam list-open-id-connect-provider-tags
- AWS API: ListOpenIDConnectProviderTags

Untuk melampirkan tag ke penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-open-id-connect-provider
- AWS API: TagOpenIDConnectProvider

Untuk menghapus tag dari penyedia identitas IAM OIDC (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-open-id-connect-provider
- AWS API: UntagOpenIDConnectProvider

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Kata kunci IAM SAML identity provider

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke penyedia identitas SAML. Misalnya, untuk mengidentifikasi penyedia, Anda dapat menambahkan kunci tanda **`okta`** dan nilai tanda **`saml`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Izin yang diperlukan untuk menandai penyedia identitas SAML

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai Penyedia Identitas berbasis SAML 2.0 (). IdPs Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Untuk mengizinkan entitas IAM (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk penyedia identitas SAML

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<SAMLProviderName>`* dengan nama penyedia SAML yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
```

```
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke penyedia identitas SAML tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk penyedia SAML tertentu.

> **Note**
>
> Tindakan `iam:TagSAMLProvider` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListSAMLProviderTags`

Untuk menggunakan kebijakan ini, ganti *<SAMLProviderName>* dengan nama penyedia SAML yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada penyedia identitas IAM SAML (konsol)

Anda dapat mengelola tag untuk Penyedia Identitas IAM SAML dari. Konsol Manajemen AWS

Untuk mengelola tanda pada penyedia identitas SAML (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Penyedia identitas, kemudian pilih nama penyedia identitas SAML yang ingin Anda edit.
3. Pilih tab Tag, lalu di bagian Tag, pilih Kel ola tag, lalu selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tanda jika penyedia identitas SAML belum memiliki tanda atau untuk menambahkan tanda baru.
   - Mengedit kunci dan nilai tanda yang ada.
   - Untuk menghapus tanda, pilih Hapus tanda.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada penyedia identitas IAM SAML (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk penyedia identitas IAM SAML. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk penyedia identitas IAM SAML.

Untuk mencantumkan tag yang saat ini dilampirkan ke penyedia identitas SAML (AWS CLI atau AWS API)

- AWS CLI: aws iam list-saml-provider-tags
- AWS API: ListSAMLProviderTags

Untuk melampirkan tag ke penyedia identitas SAML (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-saml-provider
- AWS API: TagsAmlProvider

Untuk menghapus tag dari penyedia identitas SAML (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-saml-provider
- AWS API: UntagsAmlProvider

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Tandai profil instans untuk peran Amazon EC2

Saat meluncurkan instans Amazon EC2, Anda menentukan peran IAM untuk dikaitkan dengan instans tersebut. Profil instans adalah kontainer untuk IAM role yang dapat Anda gunakan untuk meneruskan informasi peran ke instans Amazon EC2 saat instans diluncurkan. Anda dapat menandai profil instance saat menggunakan AWS API AWS CLI or.

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke profil instance. Misalnya, untuk menambahkan informasi departemen ke profil instans, Anda dapat menambahkan kunci tanda **`access-team`** dan nilai tanda **`eng`**. Melakukan hal ini memberikan prinsipal dengan akses tanda yang sesuai ke profil instans dengan tanda yang sama. Anda bisa menggunakan beberapa tag pasangan kunci-nilai tanda untuk menentukan tim dan proyek: **`access-team = eng`** , dan **`project = peg`**. Anda dapat menggunakan tanda untuk mengontrol akses pengguna ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke pengguna. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin yang diperlukan untuk menandai profil instans

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai profil instance. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Untuk mengizinkan entitas IAM (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk profil instance

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<InstanceProfileName>* dengan nama profil instans yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke profil instance tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk profil instance tertentu.

> **Note**
>
> Tindakan `iam:TagInstanceProfile` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListInstanceProfileTags`

Untuk menggunakan kebijakan ini, ganti *<InstanceProfileName>* dengan nama profil instans yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada profil instance (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk profil instans. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag misalnya profil.

Untuk mencantumkan tag yang saat ini dilampirkan ke profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam list-instance-profile-tags
- AWS API: ListInstanceProfileTags

Untuk melampirkan tag ke profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-instance-profile
- AWS API: TagInstanceProfile

Untuk menghapus tag dari profil instance (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-instance-profile
- AWS API: UntagInstanceProfile

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Sertifikat server tag

Jika Anda menggunakan IAM untuk mengelola SSL/TLS sertifikat, Anda dapat menandai sertifikat server di IAM menggunakan API AWS CLI or AWS . Untuk sertifikat di Wilayah yang didukung oleh AWS Certificate Manager (ACM), sebaiknya gunakan ACM alih-alih IAM untuk menyediakan, mengelola, dan menyebarkan sertifikat server Anda. Di Wilayah yang tidak didukung, Anda harus menggunakan IAM sebagai manajer sertifikat. Untuk mempelajari Wilayah mana yang didukung ACM, lihat AWS Certificate Manager titik akhir dan kuota di. Referensi Umum AWS

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke sertifikat server. Misalnya, untuk menambahkan informasi tentang pemilik atau administrator sertifikat server, tambahkan kunci tanda **`owner`** dan nilai tanda **`net-eng`**. Atau Anda dapat menentukan pusat biaya dengan menambahkan kunci tanda **`CostCenter`** dan nilai tanda **`1234`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke sumber daya. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin yang diperlukan untuk menandai sertifikat server

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai sertifikat server. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Untuk mengizinkan entitas IAM (pengguna atau peran) menambah, mencantumkan, atau menghapus tag untuk sertifikat server

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *`<CertificateName>`* dengan nama sertifikat server yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
```

```
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke sertifikat server tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk sertifikat server tertentu.

> **Note**
>
> Tindakan `iam:TagServerCertificate` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListServerCertificateTags`

Untuk menggunakan kebijakan ini, ganti *<CertificateName>* dengan nama sertifikat server yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada sertifikat server (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk sertifikat server. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk sertifikat server.

Untuk mencantumkan tag yang saat ini dilampirkan ke sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam list-server-certificate-tags
- AWS API: ListServerCertificateTags

Untuk melampirkan tag ke sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-server-certificate
- AWS API: TagServerCertificate

Untuk menghapus tag dari sertifikat server (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-server-certificate
- AWS API: UntagServerCertificate

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Menandai perangkat MFA virtual

Anda dapat menggunakan pasangan nilai kunci tag IAM untuk menambahkan atribut khusus ke perangkat MFA virtual. Misalnya, untuk menambahkan informasi pusat biaya untuk perangkat MFA virtual pengguna, Anda dapat menambahkan kunci tanda **`CostCenter`** dan nilai tanda **`1234`**. Anda dapat menggunakan tanda untuk mengontrol akses ke sumber daya atau untuk mengontrol tanda yang dapat dilampirkan ke objek. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Anda juga dapat menggunakan tag AWS STS untuk menambahkan atribut khusus saat Anda mengambil peran atau menyatukan pengguna. Untuk informasi selengkapnya, lihat Lewati tag sesi di AWS STS.

Izin yang diperlukan untuk menandai perangkat MFA virtual

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk menandai perangkat MFA virtual. Anda dapat menentukan satu atau semua tindakan tag IAM berikut dalam kebijakan IAM:

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`

- `iam:UntagMFADevice`

Untuk mengizinkan entitas IAM (pengguna atau peran) menambah, membuat daftar, atau menghapus tag untuk perangkat MFA virtual

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun Anda dan ganti *<MFATokenID>* dengan nama perangkat MFA virtual yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Untuk mengizinkan entitas IAM (pengguna atau peran) menambahkan tag ke perangkat MFA virtual tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk perangkat MFA tertentu.

> **Note**
>
> Tindakan `iam:TagMFADevice` mengharuskan Anda untuk juga menyertakan tindakan `iam:ListMFADeviceTags`

Untuk menggunakan kebijakan ini, ganti *<MFATokenID>* dengan nama perangkat MFA virtual yang tagnya perlu dikelola. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Atau, Anda dapat menggunakan kebijakan AWS terkelola seperti IAMFullAccess untuk menyediakan akses penuh ke IAM.

Mengelola tag pada perangkat MFA virtual (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tanda untuk perangkat MFA virtual. Anda dapat menggunakan AWS CLI atau AWS API untuk mengelola tag untuk perangkat MFA virtual.

Untuk mencantumkan tag yang saat ini dilampirkan ke perangkat MFA virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam list-mfa-device-tags
- AWS API: ListMFADeviceTags

Untuk melampirkan tag ke perangkat MFA virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam tag-mfa-device
- AWS API: TagMfaDevice

Untuk menghapus tag dari perangkat MFA virtual (AWS CLI atau AWS API)

- AWS CLI: aws iam untag-mfa-device
- AWS API: UntagmfaDevice

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat Elemen kebijakan IAM: Variabel dan tag.

Tandai instans manajer akses akun

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (peran atau pengguna) menandai instans manajer akses akun. Anda dapat menentukan satu atau semua tindakan tag akses akun berikut dalam kebijakan IAM:

- `account-access:TagResource`
- `account-access:ListTagsForResource`
- `account-access:UntagResource`

Untuk mengizinkan entitas IAM (peran atau pengguna) menambah, mencantumkan, atau menghapus tag untuk instance manajer akses akun

Tambahkan pernyataan berikut ini ke kebijakan izin untuk entitas IAM yang perlu mengelola tag. Gunakan nomor akun dan ID instans dari Pusat Identitas IAM dan instans manajer akses akun untuk mengganti placeholder. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Buat kebijakan IAM (konsol)”.

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource",
        "account-access:UntagResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Untuk mengizinkan entitas IAM (peran atau pengguna) menambahkan tag ke instance pengelola akses akun tertentu

Tambahkan pernyataan berikut ke kebijakan izin untuk entitas IAM yang perlu menambahkan, tetapi tidak menghapus, tag untuk instance manajer akses akun tertentu.

> **Note**
>
> Tindakan `account-access:TagResource` mengharuskan Anda untuk juga menyertakan tindakan `account-access:ListTagsForResource`

Gunakan nomor akun dan ID instans dari Pusat Identitas IAM dan instans manajer akses akun untuk mengganti placeholder. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Buat kebijakan IAM (konsol)”.

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Mengelola tag pada manajer akses akun (konsol)

Anda dapat mengelola tag untuk instance manajer akses akun dari Konsol Manajemen AWS.

Untuk mengelola tag pada manajer akses akun (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol, pilih Manajer akses akun, lalu pilih tab Pengaturan.
3. Di bagian Tag, Anda dapat melihat tag yang dikonfigurasi saat ini. Pilih Kelola tag lalu selesaikan salah satu tindakan berikut:
   - Pilih Tambahkan tag baru untuk menambahkan tag baru.
   - Mengedit kunci dan nilai tanda yang ada.
   - Pilih Hapus untuk menghapus tag.
4. Tambahkan atau hapus tag untuk menyelesaikan rangkaian tag. Lalu pilih Simpan Perubahan.

Mengelola tag pada manajer akses akun (AWS CLI atau AWS API)

Anda dapat membuat daftar, melampirkan, atau menghapus tag untuk instans manajer akses akun menggunakan AWS API AWS CLI or.

Untuk membuat daftar tag yang saat ini dilampirkan ke instance manajer akses akun (AWS CLI atau AWS API)

- AWS CLI: `aws account-access list-tags-for-resource`

- AWS API: `account-access:ListTagsForResource`

Untuk melampirkan tag ke instance manajer akses akun (AWS CLI atau AWS API)

- AWS CLI: `aws account-access tag-resource`
- AWS API: `TagResource`

Untuk menghapus tag dari instance manajer akses akun (AWS CLI atau AWS API)

- AWS CLI: `aws account-access untag-resource`
- AWS API: `UntagResource`

Untuk informasi tentang melampirkan tag ke sumber daya untuk AWS layanan lain, lihat dokumentasi untuk layanan tersebut.

Untuk informasi tentang menggunakan tag untuk mengatur lebih banyak izin granular dengan kebijakan izin, lihat the section called “Variabel dan tanda”.

Lewati tag sesi di AWS STS

Tag sesi adalah atribut pasangan kunci-nilai yang Anda berikan saat Anda mengambil peran IAM atau menyatukan pengguna. AWS STS Anda melakukan ini dengan membuat permintaan AWS API melalui AWS CLI AWS STS atau melalui penyedia identitas Anda (IdP). Saat Anda menggunakan AWS STS untuk meminta kredentif keamanan sementara, Anda membuat sesi. Sesi berakhir dan memiliki kredensial, seperti pasangan access key dan token sesi. Saat Anda menggunakan kredensial sesi untuk membuat permintaan berikutnya, konteks permintaan mencakup kunci konteks `aws:PrincipalTag`. Anda dapat menggunakan kunci `aws:PrincipalTag` dalam elemen `Condition` elemen kebijakan Anda untuk mengizinkan atau menolak akses berdasarkan tag tersebut.

Saat Anda menggunakan kredensial sementara untuk membuat permintaan, prinsipal Anda mungkin mencakup serangkaian tag. Tag ini berasal dari sumber berikut:

1. Tag sesi — Tag yang diteruskan saat Anda mengambil peran atau menyatukan pengguna menggunakan AWS API AWS CLI or. Untuk informasi selengkapnya tentang operasi ini, lihat Operasi pemberian tag ke sesi.

2. Tag sesi transitif masuk — Tag yang diwarisi dari sesi sebelumnya dalam rantai peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi dalam topik ini.
3. Tag IAM — Tag yang dilampirkan pada peran yang diasumsikan IAM Anda.

Topik

- Operasi pemberian tag ke sesi
- Hal yang perlu diketahui tentang tag sesi
- Izin diperlukan untuk menambahkan tag sesi
- Melewati tag sesi menggunakan AssumeRole
- Melewati tag sesi menggunakan AssumeRoleWithSAML
- Melewati tag sesi menggunakan AssumeRoleWithWebIdentity
- Melewati tag sesi menggunakan GetFederationToken
- Merangkai peran dengan tag sesi
- Menggunakan tag sesi untuk ABAC
- Melihat tag sesi di CloudTrail

Operasi pemberian tag ke sesi

Anda dapat meneruskan tag sesi menggunakan operasi berikut AWS CLI atau AWS API di AWS STS. Fitur Konsol Manajemen AWS Beralih Peran tidak memungkinkan Anda untuk meneruskan tag sesi.

Anda juga dapat mengatur tag sesi sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Tabel berikut membandingkan metode untuk meneruskan tag sesi.

Operasi	Siapa yang bisa mengambil peran	Metode untuk meneruskan tag	Metode untuk mengatur tag transitif
Operasi `assume-role` CLI atau `AssumeRole` API	Pengguna IAM atau sesi	`Tags` Parameter API atau `--tags` Opsi CLI	`TransitiveTagKeys` Parameter API

Operasi	Siapa yang bisa mengambil peran	Metode untuk meneruskan tag	Metode untuk mengatur tag transitif
			atau `--transitive-tag-keys` Opsi CLI
Operasi `assume-role-with-saml` CLI atau `AssumeRoleWithSAML` API	Setiap pengguna yang diautentikasi menggunakan penyedia identitas SAML	`PrincipalTag` Atribut SAML	`TransitiveTagKeys` Atribut SAML
Operasi `assume-role-with-web-identity` CLI atau `AssumeRoleWithWebIdentity` API	Setiap pengguna yang diautentikasi menggunakan penyedia OIDC	`PrincipalTag` Token OIDC	`TransitiveTagKeys` Token OIDC
Operasi `get-federation-token` CLI atau `GetFederationToken` API	Pengguna IAM atau pengguna root	`Tags` Parameter API atau `--tags` Opsi CLI	Tidak didukung

Operasi yang mendukung penandaan sesi dapat gagal dalam kondisi berikut:

- Anda meneruskan lebih dari 50 tag sesi.
- Teks plaintext dari kunci tag sesi Anda melebihi 128 karakter.
- Teks biasa dari nilai tag sesi melebihi 256 karakter.
- Ukuran total plaintext kebijakan sesi melebihi 2048 karakter.
- Kebijakan sesi dan tag sesi menambah ukuran token sesi. Jika token melebihi batas ukuran, operasi gagal. Pesan kesalahan melaporkan ukuran token sesi Anda dan ukuran maksimum yang

diizinkan, keduanya dalam byte. Pan `SessionTokenUtilization` tau resp AWS STS on API untuk melacak penggunaan Anda.

Hal yang perlu diketahui tentang tag sesi

Sebelum menggunakan tag sesi, tinjau detail berikut tentang sesi dan tag.

- Saat menggunakan tag sesi, kebijakan kepercayaan untuk semua peran yang terhubung ke tag penerus penyedia identitas (IdP) harus memiliki `sts:TagSession` izin. Untuk peran tanpa izin ini dalam kebijakan kepercayaan, `AssumeRole` operasi gagal.
- Saat meminta sesi, Anda dapat menentukan tag utama sebagai tag sesi. Tag berlaku untuk permintaan yang Anda buat dengan menggunakan kredensial sesi.
- Tag sesi menggunakan pasangan kunci-nilai. Misalnya, untuk menambahkan informasi kontak ke sesi, Anda dapat menambahkan kunci tag sesi `email` dan nilai tag `johndoe@example.com`.
- Tag sesi harus mengikuti aturan untuk penamaan tag di IAM dan AWS STS. Topik ini mencakup informasi tentang kepekaan huruf besar-kecil dan awalan terbatas yang berlaku untuk tag sesi Anda.
- Tag sesi baru menggantikan peran yang diasumsikan yang ada atau tag sesi pengguna federasi dengan kunci tag yang sama, terlepas dari kasus karakter.
- Anda tidak dapat meneruskan tag sesi menggunakan Konsol Manajemen AWS.
- Tag sesi hanya valid untuk sesi saat ini.
- Tag sesi mendukung perangkaian peran. Secara default, AWS STS tidak meneruskan tag ke sesi peran berikutnya. Anda juga dapat mengatur tag sesi sebagai transitif. Tag transitif tetap ada selama rantai peran dan mengganti `ResourceTag` nilai yang cocok setelah evaluasi kebijakan kepercayaan peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.
- Anda dapat menggunakan tag sesi untuk mengontrol akses ke sumber daya atau untuk mengontrol tag apa yang dapat diteruskan ke sesi selanjutnya. Untuk informasi selengkapnya, lihat Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC.
- Anda dapat melihat tag utama untuk sesi Anda, termasuk tag sesi, di AWS CloudTrail log. Untuk informasi selengkapnya, lihat Melihat tag sesi di CloudTrail.
- Anda harus meneruskan satu nilai untuk setiap tag sesi. AWS STS tidak mendukung tag sesi multi-nilai.
- Anda dapat meneruskan maksimum 50 tag sesi. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

- Kebijakan sesi dan tag sesi menambah ukuran token sesi. Jika token melebihi batas ukuran, AWS API AWS CLI or mengembalikan kesalahan. Pesan kesalahan melaporkan ukuran token sesi Anda dan ukuran maksimum yang diizinkan, keduanya dalam byte. Elem `SessionTokenUtilization` en respons menunjukkan berapa persentase dari batas ukuran yang digunakan token. Elem `PackedPolicySize` en respons tidak digunakan lagi. Gunakan `SessionTokenUtilization` sebagai gantinya.

Izin diperlukan untuk menambahkan tag sesi

Selain tidakan yang sesuai dengan operasi API, Anda harus memiliki tindakan khusus izin berikut dalam kebijakan Anda:

```
sts:TagSession
```

> **Important**
>
> Saat menggunakan tag sesi, kebijakan kepercayaan peran untuk semua peran yang terhubung ke penyedia identitas (IdP) harus memiliki izin `sts:TagSession`. `AssumeRole`Operasi gagal untuk peran apa pun yang terhubung ke IdP yang melewati tag sesi tanpa izin ini. Jika Anda tidak ingin memperbarui kebijakan kepercayaan peran untuk setiap peran, Anda dapat menggunakan instans IdP terpisah untuk meneruskan tag sesi. Kemudian, tambahkan `sts:TagSession` izin hanya untuk peran yang terhubung ke IdP terpisah.

Anda dapat menggunakan tindakan `sts:TagSession` dengan kunci syarat berikut.

- `aws:PrincipalTag`— Membandingkan tag yang dilampirkan pada kepala sekolah yang membuat permintaan dengan tag yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi hanya jika prinsipal yang mengajukan permintaan memiliki tag yang ditentukan.
- `aws:RequestTag`— Membandingkan pasangan nilai kunci tag yang diteruskan dalam permintaan dengan pasangan tag yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk meneruskan tag sesi yang ditentukan, tetapi hanya dengan nilai yang ditentukan.
- `aws:ResourceTag`— Membandingkan pasangan nilai kunci tag yang Anda tentukan dalam kebijakan dengan pasangan kunci-nilai yang dilampirkan ke sumber daya. Misalnya, Anda dapat

mengizinkan prinsipal untuk meneruskan tag sesi hanya jika peran yang mereka asumsikan mencakup tag yang ditentukan.

- `aws:TagKeys`— Membandingkan kunci tag dalam permintaan dengan kunci yang Anda tentukan dalam kebijakan. Misalnya, Anda dapat mengizinkan prinsipal untuk hanya meneruskan tag sesi dengan kunci tag yang ditentukan. Kunci persyaratan ini membatasi rangkaian tag sesi maksimum yang dapat diteruskan.
- `sts:TransitiveTagKeys`- Membandingkan kunci tag sesi transitif dalam permintaan dengan yang ditentukan dalam kebijakan. Misalnya, Anda dapat menyusun kebijakan untuk mengizinkan prinsipal mengatur hanya tag tertentu sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Misalnya, kebijakan kepercayaan peran berikut memungkinkan `test-session-tags` pengguna untuk mengambil peran dengan kebijakan terlampir. Ketika pengguna tersebut mengambil peran, mereka harus menggunakan AWS API AWS CLI atau untuk meneruskan tiga tag sesi yang diperlukan dan ID eksternal yang diperlukan. Selain itu, pengguna dapat memilih untuk mengatur tag `Project` dan `Department` tag sebagai transitif.

Example Contoh kebijakan kepercayaan peran untuk tag sesi

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
```

```
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

Apa yang dilakukan kebijakan ini?

- Per `AllowIamUserAssumeRole` nyataan ini memungkinkan `test-session-tags` pengguna untuk mengambil peran dengan kebijakan terlampir. Ketika pengguna tersebut mengasumsikan peran tersebut, mereka harus meneruskan tag sesi yang diperlukan dan ID eksternal.
  - Blok syarat pertama dari pernyataan ini mengharuskan pengguna untuk meneruskan `Project`, `CostCenter`, dan `Department` tag sesi. Nilai tag tidak penting dalam pernyataan ini, jadi Anda dapat menggunakan wildcard (*) untuk nilai tag. Blok ini memastikan bahwa pengguna melewati setidaknya tiga tag sesi ini. Jika tidak, operasi gagal. Pengguna dapat meneruskan tag tambahan.
  - Blok syarat kedua mengharuskan pengguna untuk meneruskan ID eksternal dengan nilai `Example987`.

- Pernyataan `AllowPassSessionTagsAndTransitive` memperbolehkan tindakan khusus izin `sts:TagSession`. Tindakan ini harus diperbolehkan sebelum pengguna dapat meneruskan tag sesi. Jika kebijakan Anda mencakup pernyataan pertama tanpa pernyataan kedua, pengguna tidak dapat mengasumsikan peran tersebut.
  - Blok syarat pertama dari pernyataan ini memperbolehkan pengguna untuk meneruskan nilai apa pun untuk sesi tag `CostCenter` dan `Project`. Anda melakukannya dengan menggunakan wildcards (*) untuk nilai tag dalam kebijakan, yang mengharuskan Anda menggunakan operator syarat StringLike.
  - Blok syarat kedua memperbolehkan pengguna untuk hanya meneruskan `Engineering` atau `Marketing` nilai untuk tag sesi `Department`.
  - Blok kondisi ketiga mencantumkan set tag maksimum yang dapat Anda tetapkan sebagai transitif. Pengguna dapat memilih untuk mengatur subset atau tidak ada tag sebagai transitif. Mereka tidak dapat menetapkan tag tambahan sebagai transitif. Anda dapat meminta agar mereka mengatur setidaknya salah satu tag sebagai transtitif dengan menambahkan blok syarat lain yang mencakup `"Null":{"sts:TransitiveTagKeys":"false"}`.

Melewati tag sesi menggunakan AssumeRole

`AssumeRole`Operasi mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Anda dapat menggunakan kredentif pengguna atau peran IAM untuk memanggil`AssumeRole`. Untuk meneruskan tag sesi sambil mengambil peran, gunakan `--tags` AWS CLI opsi atau parameter `Tags` AWS API.

Untuk mengatur tag sebagai transitif, gunakan `--transitive-tag-keys` AWS CLI opsi atau parameter `TransitiveTagKeys` AWS API. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Contoh berikut menunjukkan permintaan sampel yang menggunakan `AssumeRole`. Dalam contoh ini, saat mengambil peran `my-role-example`, Anda membuat sesi bernama `my-session`. Anda menambahkan pasangan nilai kunci tanda sesi `Project` = `Automation`, `CostCenter` = `12345`, dan `Department` = `Engineering`. Anda juga mengatur tanda `Project` dan `Department` sebagai transitif dengan menentukan kuncinya. Anda harus meneruskan satu nilai untuk setiap tag sesi. AWS STS tidak mendukung tag sesi multi-nilai.

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Melewati tag sesi menggunakan AssumeRoleWithSAML

`AssumeRoleWithSAML`Operasi diautentikasi dengan SAML-based federasi. Operasi ini mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang menggunakan SAML-based federasi untuk Konsol Manajemen AWS akses, lihatMengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS. Untuk detail tentang AWS CLI atau akses AWS API, lihatFederasi SAML 2.0. Untuk tutorial tentang mengonfigurasi federasi SAML untuk pengguna Active Directory Anda, lihat Otentikasi AWS Federasi dengan Layanan Federasi Direktori Aktif (ADFS) di Blog Keamanan. AWS

Sebagai administrator, Anda dapat mengizinkan anggota direktori perusahaan Anda untuk bergabung AWS menggunakan AWS STS `AssumeRoleWithSAML` operasi. Untuk melakukannya, Anda harus menyelesaikan tugas berikut:

1. Konfigurasikan jaringan Anda sebagai penyedia SAML untuk AWS.
2. Buat penyedia SAML di IAM
3. Buat peran untuk federasi SAML 2.0 (konsol)
4. Selesaikan konfigurasi SAML IdP dan buat pernyataan untuk respons otentikasi SAML

AWS mencakup penyedia identitas dengan pengalaman end-to-end bersertifikat untuk tag sesi dengan solusi identitas mereka. Untuk mempelajari cara menggunakan penyedia identitas ini untuk mengonfigurasi tag sesi, lihat Integrasikan penyedia solusi SAML pihak ketiga dengan AWS.

Untuk meneruskan atribut SAML sebagai tanda sesi, sertakan elemen `Attribute` dengan atribut `Name` yang diatur ke `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`*. Gunakan elemen `AttributeValue` untuk menentukan nilai tanda. Menyertakan elemen `Attribute` terpisah untuk setiap tag sesi.

Misalnya, anggap bahwa Anda ingin meneruskan atribut identitas berikut sebagai tag sesi:

- `Project:Automation`

- `CostCenter:12345`
- `Department:Engineering`

Untuk meneruskan atribut ini, sertakan elemen berikut ini dalam pernyataan SAML Anda.

Example Contoh snippet pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Untuk mengatur tag sebelumnya sebagai transitif, sertakan `Attribute` elemen lain dengan `Name` atribut yang disetel ke`https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Untuk mengatur `Department` tag `Project` and sebagai transitif, gunakan atribut multi-nilai berikut:

Example Contoh snippet pernyataan SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Melewati tag sesi menggunakan AssumeRoleWithWebIdentity

Gunakan federasi yang sesuai dengan OpenID Connect (OIDC) untuk mengotentikasi operasi. `AssumeRoleWithWebIdentity` Operasi ini mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk informasi selengkapnya tentang menggunakan federasi identitas web untuk Konsol Manajemen AWS akses, lihatFederasi OIDC.

Untuk meneruskan tag sesi dari OpenID Connect (OIDC), Anda harus menyertakan tag sesi di Token Web JSON (JWT) saat Anda mengirimkan permintaan. `AssumeRoleWithWebIdentity` Untuk

mempelajari lebih lanjut tentang token dan klaim OIDC, lihat Menggunakan Token dengan Kolam Pengguna di Panduan Developer Amazon Cognito .

AWS mendukung dua format klaim untuk memasukkan tag sesi di JWT:

- Format klaim bersarang
- Format klaim yang diratakan

Format klaim bersarang

Format klaim bersarang menggunakan struktur dalam `https://aws.amazon.com/tags` namespace di JWT. Dalam format ini:

- Tag utama direpresentasikan sebagai objek bersarang di bawah `principal_tags` kunci.
- Setiap tag utama adalah nilai string tunggal.
- Kunci tag transitif diwakili dalam array di bawah `transitive_tag_keys` kunci.
- Kedu `principal_tags` anya dan `transitive_tag_keys` bersarang di bawah `https://aws.amazon.com/tags` namespace.

Contoh berikut menunjukkan JWT yang didekodekan menggunakan format objek bersarang:

Example Contoh Token Web JSON yang didekodekan menggunakan format klaim bersarang

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
```

```
            "CostCenter"
        ]
    }
}
```

Format klaim yang diratakan

Format klaim yang diratakan kompatibel dengan penyedia identitas yang tidak mendukung objek bersarang dalam klaim JWT, seperti Microsoft Entra ID. Dalam format ini:

- Tag utama direpresentasikan sebagai klaim terpisah dengan awalan`https://aws.amazon.com/tags/principal_tags/`.
- Setiap tag utama adalah nilai string tunggal.
- Kunci tag transitif direpresentasikan dalam satu klaim sebagai array string dengan aw `https://aws.amazon.com/tags/transitive_tag_keys` alan.

Sekarang, mari kita lihat bagaimana informasi yang sama direpresentasikan menggunakan format klaim yang diratakan:

Example Contoh Token Web JSON yang didekodekan menggunakan format klaim yang diratakan

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags/principal_tags/Project": "Automation",
    "https://aws.amazon.com/tags/principal_tags/CostCenter": "987654",
    "https://aws.amazon.com/tags/principal_tags/Department": "Engineering",
    "https://aws.amazon.com/tags/transitive_tag_keys": [
        "Project",
        "CostCenter"
    ]
}
```

Kedua contoh JWT yang didekodekan menunjukkan panggilan ke `AssumeRoleWithWebIdentity` dengan tag`Project`,`CostCenter`, dan `Department` sesi. Kedua token menetapkan `CostCenter`

tag `Project` dan sebagai transitif. Tag transitif tetap ada selama perangkaian peran. Untuk informasi selengkapnya, lihat Merangkai peran dengan tag sesi.

Format klaim yang diratakan mencapai hasil yang sama dengan format klaim bersarang tetapi menggunakan struktur yang diratakan untuk tag. Ini memungkinkan Anda untuk memasukkan tag sesi di lingkungan di mana objek JSON bersarang tidak didukung dalam klaim JWT. Saat menggunakan salah satu format, pastikan penyedia identitas Anda dikonfigurasi untuk mengeluarkan token dengan struktur klaim yang sesuai. AWS mendukung kedua format klaim, sehingga Anda dapat memilih salah satu yang paling sesuai dengan persyaratan spesifik penyedia identitas Anda.

Melewati tag sesi menggunakan GetFederationToken

Ini `GetFederationToken` memungkinkan Anda untuk menyatukan pengguna Anda. Operasi ini mengembalikan satu set kredenSIAL sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya. Untuk menambahkan tag ke sesi pengguna federasi Anda, gunakan `--tags` AWS CLI opsi atau parameter `Tags` AWS API. Anda tidak dapat menetapkan tag sesi sebagai transitif saat menggunakan`GetFederationToken`, karena Anda tidak dapat menggunakan kredenSIAL sementara untuk mengambil peran. Anda tidak dapat menggunakan rantai peran dalam kasus ini.

Contoh berikut menunjukkan permintaan sampel menggunakan `GetFederationToken`. Dalam contoh ini, saat meminta token, Anda membuat sesi bernama `my-fed-user`. Anda menambahkan pasangan nilai kunci tanda sesi `Project` = `Automation` dan `Department` = `Engineering`.

Example Contoh GetFederationToken permintaan CLI

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Saat Anda menggunakan kredenSIAL sementara yang dikembalikan oleh `GetFederationToken` operasi, tag utama sesi menyertakan tag pengguna dan tag sesi yang dilewati.

Merangkai peran dengan tag sesi

Anda dapat mengasumsikan satu peran dan kemudian menggunakan kredensial sementara untuk mengasumsikan peran lain. Anda dapat melanjutkan dari sesi ke sesi. Ini disebut perangkaian peran. Saat Anda meneruskan tag sesi sembari mengasumsikan peran, Anda dapat mengatur kunci sebagai transitif. Hal ini memastikan bahwa tag sesi tersebut diteruskan ke sesi berikutnya dalam rantai peran. Anda tidak dapat mengatur tag peran sebagai transitif. Untuk meneruskan tag ini ke sesi berikutnya, tentukan tag tersebut sebagai tag sesi.

> **Note**
>
> Tag transitif tetap ada selama rantai peran dan mengganti `ResourceTag` nilai yang cocok setelah evaluasi kebijakan kepercayaan peran.

Contoh berikut menunjukkan bagaimana mener AWS STS uskan tag sesi, tag transitif, dan tag peran ke sesi berikutnya dalam rantai peran.

Dalam contoh skenario rantai peran ini, Anda menggunakan kunci akses pengguna IAM AWS CLI untuk mengambil peran bernama. `Role1` Anda kemudian menggunakan kredensial sesi yang dihasilkan untuk mengambil peran kedua bernama `Role2`. Anda kemudian dapat menggunakan kredensial sesi kedua untuk mengambil peran ketiga bernama `Role3`. Permintaan ini terjadi sebagai tiga operasi terpisah. Setiap peran sudah diberi tag di IAM. Dan dalam setiap permintaan, Anda meneruskan tag sesi tambahan.
Rantaian peran.

Saat Anda merangkai peran, Anda dapat memastikan bahwa tag dari sesi sebelumnya tetap berlanjut ke sesi berikutnya. Untuk melakukan ini menggunakan perintah `assume-role` CLI, Anda harus meneruskan tag sebagai tag sesi dan mengatur tag sebagai transitif. Anda meneruskan tag `Star = 1` sebagai tag sesi. Perintah juga melampirkan tag `Heart = 1` ke peran dan berlaku sebagai tag utama saat Anda menggunakan sesi. Namun, Anda juga menginginkan tag `Heart = 1` tag untuk diteruskan secara otomatis ke sesi kedua atau ketiga. Untuk melakukannya, Anda menyertakannya secara manual sebagai tag sesi. Tag utama sesi yang dihasilkan mencakup kedua tag ini, dan menetapkannya sebagai transitif.
Mengasumsikan peran pertama dalam rantai peran.

Anda melakukan permintaan ini menggunakan AWS CLI perintah berikut:

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Anda kemudian menggunakan kredensials untuk sesi tersebut guna mengambil `Role2`. Perintah melampirkan tag `Sun = 2` ke peran kedua dan berlaku sebagai tag utama saat Anda menggunakan

sesi kedua. StarTag Heart and mewarisi tag sesi transitif di sesi pertama. Sesi kedua yang menghasilkan tag utama adalah Heart =1, Star =1, dan Sun =2. Heartdan Star terus menjadi transitif. SunTag yang dil Role2 ampirkan tidak ditandai sebagai transitif karena bukan tag sesi. Sesi mendatang tidak mewarisi tag ini.
Mengasumsikan peran kedua dalam rantai peran.

Anda melakukan permintaan kedua ini menggunakan AWS CLI perintah berikut:

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Anda kemudian menggunakan kredensial kedua guna mengambil Role3. Tanda prinsipal untuk sesi ketiga berasal dari tanda sesi baru, tanda sesi transitif yang diturunkan, dan tanda peran. 1Tag Heart = 1 dan Star = pada sesi kedua diwarisi dari tag sesi transitif di sesi pertama. Jika Anda mencoba meneruskan tag 2 sesi Sun =, operasi gagal. Tag sesi Star = 1 yang diwariskan menggantikan 3 tag role Star =. Dalam rantai peran, nilai tag transitif menggantikan peran yang cocok dengan ResourceTag nilai setelah evaluasi kebijakan kepercayaan peran. Dalam contoh ini, Role3 jika Star digunakan sebagai ResourceTag kebijakan kepercayaan peran, dan menetapkan ResourceTag nilai ke nilai tag transitif dari sesi peran panggilan. LightningTag peran juga berlaku untuk sesi ketiga, dan tidak disetel sebagai transitif.
Mengasumsikan peran ketiga dalam rantai peran.

Anda melakukan permintaan ketiga menggunakan AWS CLI perintah berikut:

Example Contoh AssumeRole permintaan CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Menggunakan tag sesi untuk ABAC

Attribute-based kontrol akses (ABAC) menggunakan strategi otorisasi yang mendefinisikan izin berdasarkan atribut tag.

Jika perusahaan Anda menggunakan OIDC atau penyedia SAML-based identitas (IdP) untuk mengelola identitas pengguna, Anda dapat mengonfigurasi pernyataan untuk meneruskan tag sesi

ke. AWS Misalnya, dengan identitas pengguna perusahaan, ketika karyawan Anda bergabung AWS, AWS menerapkan atribut mereka ke prinsipal yang dihasilkan. Kemudian, Anda dapat menggunakan ABAC untuk mengizinkan atau menolak izin berdasarkan atribut tersebut. Lihat perinciannya di Tutorial IAM: Menggunakan tanda sesi SAML untuk ABAC.

Untuk informasi selengkapnya tentang menggunakan Pusat Identitas IAM dengan ABAC, lihat A tribut untuk kontrol akses di Panduan AWS IAM Identity Center Pengguna.

Melihat tag sesi di CloudTrail

Anda dapat menggunakan AWS CloudTrail untuk melihat permintaan yang digunakan untuk mengambil peran atau menyatukan pengguna. File log CloudTrail mencakup informasi tentang tanda prinsipal untuk sesi peran yang diasumsikan atau sesi pengguna gabungan. Untuk informasi selengkapnya, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail.

Misalnya, asumsikan bahwa Anda membuat AWS STS `AssumeRoleWithSAML` permintaan, meneruskan tag sesi, dan menetapkan tag tersebut sebagai transitif. Anda dapat menemukan informasi berikut di log CloudTrail Anda.

Example Contoh AssumeRoleWithSAML CloudTrail log

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

Anda dapat melihat contoh CloudTrail log berikut untuk melihat peristiwa yang menggunakan tag sesi.

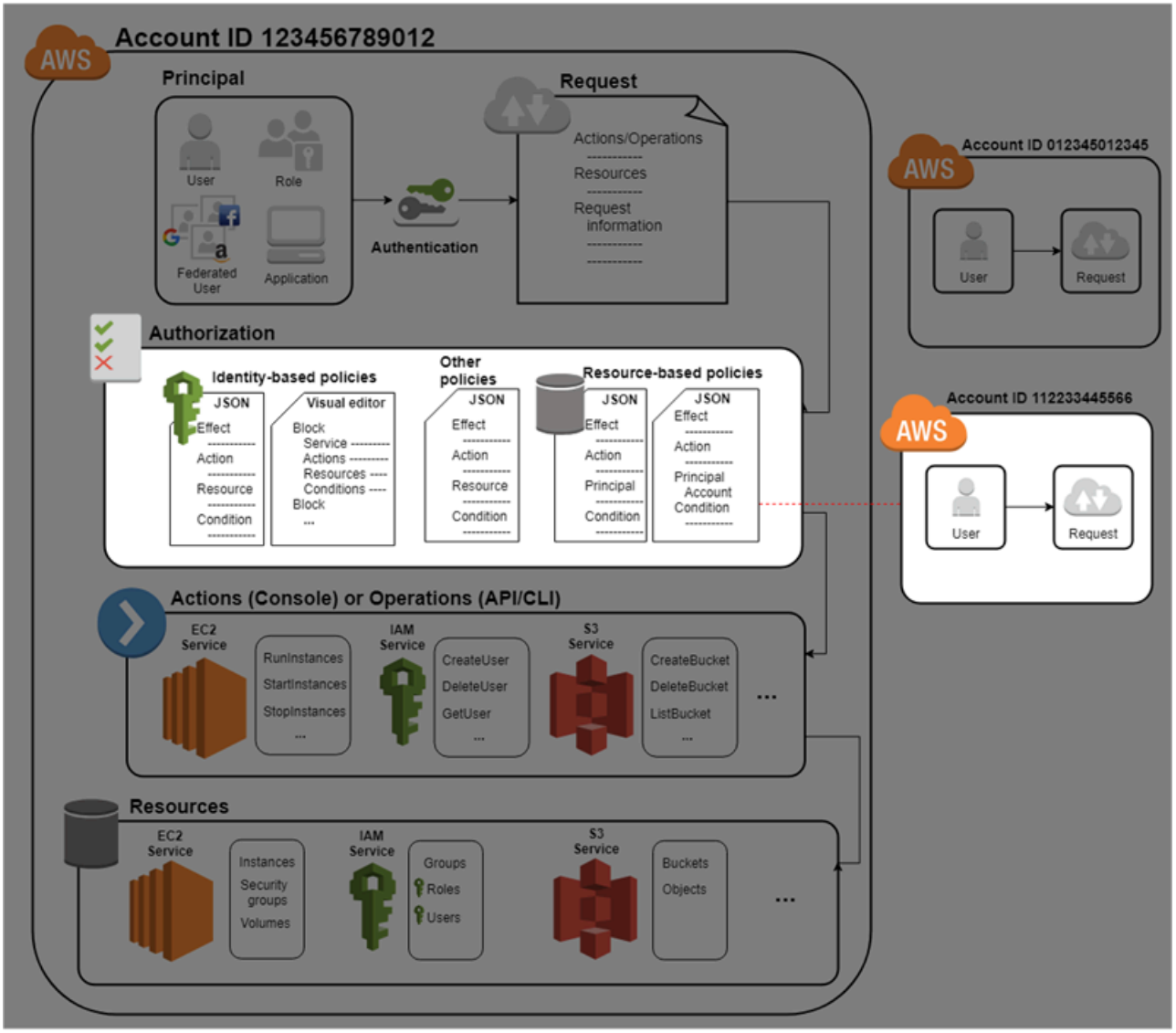
- Contoh AWS STS peristiwa API rantai peran dalam file CloudTrail log

- Contoh SAML AWS STS Peristiwa API dalam file CloudTrail log
- Contoh OIDC AWS STS Peristiwa API dalam file CloudTrail log

Manajemen akses untuk AWS sumber daya

AWS Identity and Access Management (IAM) adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman. Ketika kepala sekolah mengajukan permintaan AWS, kode AWS penegakan memeriksa apakah prinsipal diautentikasi (masuk) dan diotorisasi (memiliki izin). Anda mengelola akses AWS dengan membuat kebijakan dan melampirkannya ke identitas atau AWS sumber daya IAM. Kebijakan adalah dokumen JSON di mana AWS , ketika dilampirkan ke identitas atau sumber daya, menentukan izin mereka. Untuk informasi selengkapnya tentang jenis dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Untuk perincian tentang proses autentikasi dan otorisasi lainnya, lihat Cara kerja IAM.



Selama otorisasi, AWS kode penegakan menggunakan nilai dari konteks permintaan untuk memeriksa kebijakan yang cocok dan menentukan apakah akan mengizinkan atau menolak permintaan.

AWS memeriksa setiap kebijakan yang berlaku untuk konteks permintaan. Jika satu kebijakan menolak permintaan, AWS menolak seluruh permintaan dan menghentikan evaluasi kebijakan. Ini disebut sebagai penolakan secara tegas. Karena permintaan ditolak secara default, IAM mengizinkan permintaan Anda hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan yang berlaku. logika evaluasi untuk permintaan dalam satu akun mengikuti aturan berikut:

- Secara default, semua permintaan ditolak secara implisit. (Atau, secara default, Pengguna root akun AWS memiliki akses penuh.)
- Izin eksplisit dalam kebijakan berbasis identitas atau berbasis sumber daya akan membatalkan pengaturan default ini.
- Jika batas izin, AWS Organizations SCP, atau kebijakan sesi ada, itu mungkin mengganti izin dengan penolakan implisit.
- Penolakan secara tegas dalam kebijakan apa pun akan mengesampingkan izin apa pun.

Setelah permintaan Anda diautentikasi dan disahkan, AWS menyetujui permintaan tersebut. Jika Anda perlu mengajukan permintaan di akun yang berbeda, kebijakan di akun lain harus memungkinkan Anda mengakses sumber daya. Selain itu, entitas IAM yang Anda gunakan untuk mengajukan permintaan harus memiliki kebijakan berbasis identitas yang memungkinkan permintaan tersebut.

Anda juga dapat menggunakannya Delegasi sementara IAM untuk memberi Amazon dan AWS Mitra akses terbatas dan terbatas waktu untuk mengonfigurasi AWS layanan di akun Anda.

Sumber daya manajemen akses

Untuk informasi selengkapnya tentang izin dan tentang pembuatan kebijakan, lihat sumber daya berikut:

Entri berikut di Blog AWS Keamanan mencakup cara-cara umum untuk menulis kebijakan untuk akses ke bucket dan objek Amazon S3.

- Menulis Kebijakan IAM: Cara Memberikan Akses ke Bucket Amazon S3
- Menulis kebijakan IAM: Berikan Akses ke User-Specific Folder di Bucket Amazon S3

- Kebijakan IAM dan Kebijakan Bucket dan ACL! Astaga! (Mengontrol Akses ke Sumber Daya S3)
- Primer tentang Izin RDS Resource-Level
- Demistifikasi Izin EC2 Resource-Level

Kebijakan dan izin di AWS Identity and Access Management

Kelola akses AWS dengan membuat kebijakan dan melampirkannya ke identitas IAM (pengguna, grup pengguna, atau peran) atau AWS sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izin mereka. AWS mengevaluasi kebijakan ini ketika kepala sekolah IAM (pengguna atau peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. AWS mendukung sembilan jenis kebijakan: kebijakan berbasis identitas, kebijakan berbasis sumber daya, kebijakan titik akhir VPC, batas izin, kebijakan kontrol AWS Organizations layanan (SCP), kebijakan kontrol AWS Organizations sumber daya (RCP), daftar kontrol akses (ACL), berbagi Resource Access Manager (RAM) dan kebijakan sesi.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, jika kebijakan mengizinkan GetUser tindakan tersebut, maka pengguna dengan kebijakan tersebut dapat memperoleh informasi pengguna dari Konsol Manajemen AWS AWS CLI,, atau AWS API. Saat Anda membuat pengguna IAM, Anda dapat memilih untuk mengizinkan akses konsol atau program. Jika akses konsol diizinkan, pengguna IAM dapat masuk ke konsol menggunakan kredentif masuk mereka. Jika akses terprogram diizinkan, pengguna dapat menggunakan kunci akses untuk bekerja dengan CLI atau API.

Jenis kebijakan

Jenis kebijakan berikut, yang tercantum dalam urutan dari yang paling sering digunakan hingga yang jarang digunakan, tersedia untuk digunakan di AWS. Untuk perincian selengkapnya, lihat bagian di bawah ini untuk setiap jenis kebijakan.

- Identity-basedkebijakan — Lampirkan kebijakan ter kelola dan sebaris ke identitas IAM (pengguna, grup tempat pengguna berada, atau peran). Identity-based kebijakan memberikan izin ke identitas.
- Resource-basedkebijakan — Lampirkan kebijakan sebaris ke sumber daya. Contoh kebijakan berbasis sumber daya yang paling umum adalah kebijakan bucket Amazon S3 dan kebijakan kepercayaan peran IAM. Resource-based kebijakan memberikan izin kepada prinsipal yang

ditentukan dalam kebijakan. Principal dapat berada di akun yang sama dengan sumber daya atau di akun lain.

- Kebijakan titik akhir VPC — Lampirkan ke titik akhir VPC untuk mengontrol prinsipal mana yang dapat menggunakan kebijakan titik akhir VPC dan sumber daya mana yang dapat diakses melaluinya. Kebijakan titik akhir VPC bertindak sebagai batas akses tambahan yang dicakup untuk lalu lintas yang melintasi titik akhir.
- Batas izin – Gunakan kebijakan terkelola sebagai batas izin untuk entitas IAM (pengguna atau peran). Kebijakan tersebut menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas, tetapi tidak memberikan izin. Batas izin tidak menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis sumber daya kepada entitas.
- AWS Organizations SCP — Gunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk menentukan izin maksimum untuk pengguna IAM dan peran IAM dalam akun di organisasi atau unit organisasi (OU) Anda. SCP membatasi izin yang diberikan kebijakan berbasis identitas atau kebijakan berbasis sumber daya kepada pengguna IAM atau peran IAM dalam akun. SCP tidak memberikan izin.
- AWS Organizations RCP — Gunakan kebijakan kontrol AWS Organizations sumber daya (RCP) untuk menentukan izin maksimum untuk sumber daya dalam akun di organisasi atau unit organisasi (OU) Anda. RCP membatasi izin yang dapat diberikan kebijakan berbasis identitas dan berbasis sumber daya ke sumber daya di akun dalam organisasi Anda. RCP tidak memberikan izin.
- Daftar kontrol akses (ACL) — Gunakan ACL untuk mengontrol prinsipal mana di akun lain yang dapat mengakses sumber daya yang dilampirkan ACL. ACL serupa dengan kebijakan berbasis sumber daya, meskipun merupakan satu-satunya jenis kebijakan yang tidak menggunakan struktur dokumen kebijakan JSON. ACL adalah kebijakan izin lintas akun yang memberikan izin kepada penanggung jawab tertentu. ACL tidak dapat memberikan izin untuk entitas dalam akun yang sama.
- AWS Berbagi sumber daya RAM — Gunakan AWS Resource Access Manager untuk berbagi sumber daya di seluruh Akun AWS, unit organisasi, atau seluruh organisasi tanpa menulis kebijakan berbasis sumber daya individual untuk setiap sumber daya bersama.
- Kebijakan sesi — Luluskan kebijakan sesi lanjutan saat Anda menggunakan AWS API AWS CLI atau untuk mengambil peran atau pengguna federasi. Kebijakan sesi membatasi izin yang diberikan oleh peran atau kebijakan berbasis identitas pengguna ke sesi. Kebijakan sesi membatasi izin untuk sesi yang dibuat, tetapi tidak memberikan izin. Untuk informasi lebih lanjut, lihat Kebijakan Sesi

Identity-based kebijakan

Identity-based kebijakan adalah dokumen kebijakan izin JSON yang mengontrol tindakan apa yang dapat dilakukan identitas (pengguna, grup pengguna, dan peran), pada sumber daya mana, dan dalam kondisi apa. Identity-based kebijakan dapat dikategorikan lebih lanjut:

- Kebijakan ter kelola — Kebijakan berbasis identitas mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Anda. Akun AWS Ada dua jenis kebijakan terkelola:
  - AWS kebijakan terkelola — Kebijakan terkelola yang dibuat dan dikelola oleh AWS.
  - Kebijakan yang dikelola pelanggan — Kebijakan terkelola yang Anda buat dan kelola di Akun AWS. Kebijakan yang dikelola pelanggan memberikan kontrol yang lebih tepat atas kebijakan Anda daripada kebijakan yang AWS dikelola.
- Kebijakan inline – Kebijakan yang Anda tambahkan secara langsung ke satu pengguna, grup, atau peran. Kebijakan inline mempertahankan hubungan satu-ke-satu yang ketat antara kebijakan dan sebuah identitas. Mereka dihapus ketika Anda menghapus identitas.

Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat Pilih antara kebijakan terkelola dan kebijakan inline.

Resource-based kebijakan

Resource-based kebijakan adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya seperti bucket Amazon S3. Kebijakan ini memberikan izin utama yang ditentukan untuk melakukan tindakan tertentu pada sumber daya tersebut dan menentukan dalam kondisi apa hal ini berlaku. Resource-basedkebijakan adalah kebijakan sebaris. Tidak ada kebijakan berbasis sumber daya terkelola.

Untuk mengaktifkan akses akun silang, Anda dapat menentukan seluruh akun atau entitas IAM di akun lain sebagai penanggung jawab kebijakan berbasis sumber daya. Menambahkan principal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsip dan sumber daya terpisah Akun AWS, Anda juga harus menggunakan kebijakan berbasis identitas untuk memberikan akses utama ke sumber daya. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk petunjuk langkah demi langkah untuk memberikan akses lintaslayanan, lihat Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM.

Layanan IAM hanya mendukung satu jenis kebijakan berbasis sumber daya yang disebut peran kebijakan kepercayaan, yang terlampir pada peran IAM. Peran IAM adalah identitas dan sumber daya yang mendukung kebijakan berbasis sumber daya. Oleh karena itu, Anda harus melampirkan baik kebijakan kepercayaan dan kebijakan berbasis identitas pada peran IAM. Kebijakan kepercayaan menentukan entitas prisipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang peran tersebut. Untuk mempelajari bagaimana peran IAM berbeda dengan kebijakan berbasis sumber daya lainnya, lihat Akses sumber daya lintas akun di IAM.

Untuk melihat mana layanan lainnya yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari lebih lanjut tentang kebijakan berbasis sumber daya, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organization atau akun terpercaya) memiliki akses untuk mengambil peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

Kebijakan VPC endpoint

Kebijakan titik akhir VPC adalah kebijakan berbasis sumber daya yang Anda lampirkan ke titik akhir VPC untuk mengontrol prinsipal mana yang dapat menggunakan titik akhir dan sumber daya mana yang dapat diakses melaluinya. Kebijakan titik akhir VPC tidak mengganti atau mengganti kebijakan berbasis identitas atau kebijakan berbasis sumber daya yang dilampirkan ke layanan tujuan—kebijakan tersebut bertindak sebagai batas akses tambahan yang dicakup untuk lalu lintas yang melintasi titik akhir.

Kebijakan titik akhir VPC digunakan untuk memastikan bahwa hanya identitas tepercaya di AWS Organizations organisasi Anda yang dapat mengakses sumber daya tepercaya melalui titik akhir VPC Anda. Jika Anda tidak melampirkan kebijakan titik akhir khusus, AWS lampirkan kebijakan default yang memungkinkan akses penuh. Untuk informasi selengkapnya tentang kebijakan titik akhir, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di AWS PrivateLink Panduan.

Batas izin IAM

Batas izin adalah fitur lanjutan di mana Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM. Saat Anda menetapkan batas izin untuk suatu entitas, entitas hanya dapat melakukan tindakan yang diizinkan oleh kedua kebijakan berbasis identitas dan batas izinnya.

Jika Anda menentukan sesi peran atau pengguna dalam elemen utama kebijakan berbasis sumber daya, izin eksplisit dalam batas izin tidak diperlukan. Namun, jika Anda menentukan peran ARN dalam elemen utama kebijakan berbasis sumber daya, izin eksplisit dalam batas izin diperlukan.

Dalam kedua kasus, penolakan eksplisit dalam batas izin efektif. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi lebih lanjut tentang batas izin, lihat Batas izin untuk entitas IAM.

AWS Organizations kebijakan kontrol layanan (SCP)

Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP adalah kebijakan JSON yang menentukan izin maksimum untuk pengguna IAM dan peran IAM dalam akun organisasi atau unit organisasi (OU). SCP membatasi izin untuk kepala sekolah di akun anggota, termasuk masing-masing. Pengguna root akun AWS Penolakan eksplisit dalam salah satu kebijakan ini mengesampingkan izin dalam kebijakan lain.

Untuk informasi selengkapnya tentang AWS Organizations dan SCP, lihat Kebijakan kontrol layanan (SCP) di Panduan AWS Organizations Pengguna.

AWS Organizations kebijakan kontrol sumber daya (RCP)

Jika Anda mengaktifkan semua fitur dalam organisasi, maka Anda dapat menggunakan kebijakan kontrol sumber daya (RCP) untuk menerapkan kontrol akses secara terpusat pada sumber daya di beberapa Akun AWS sumber daya. RCP adalah kebijakan JSON yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui kebijakan IAM yang dilampirkan ke setiap sumber daya yang Anda miliki. RCP membatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah mereka milik organisasi Anda. Penolakan eksplisit dalam RCP yang berlaku mengesampingkan izin dalam kebijakan lain yang mungkin dilampirkan ke identitas atau sumber daya individu.

Untuk informasi selengkapnya tentang AWS Organizations dan RCP termasuk daftar R Layanan AWS CP yang mendukung, lihat Kebijakan kontrol sumber daya (RCP) di Panduan Pengguna. AWS Organizations

Access Control Lists (ACLs)

Access Control Lists (ACLs) adalah kebijakan layanan yang memungkinkan Anda mengontrol prinsipal yang mana di akun lain yang dapat mengakses sumber daya. ACL tidak dapat digunakan untuk mengontrol akses bagi penanggung jawab dalam akun yang sama. ACL mirip dengan kebijakan berbasis sumber daya, meskipun ini adalah satu-satunya jenis kebijakan yang tidak menggunakan format dokumen kebijakan JSON. Amazon S3, AWS WAF, dan Amazon VPC adalah

contoh layanan yang mendukung ACL. Untuk mempelajari lebih lanjut tentang ACL, lihat ikhtis ar Daftar Kontrol Akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

AWS Manajer Akses Sumber Daya (AWS RAM) pembagian sumber daya

AWS Resource Access Manager (AWS RAM) memungkinkan Anda berbagi sumber daya yang Anda buat dalam satu Akun AWS dengan yang lain Akun AWS, unit organisasi, atau seluruh organisasi di dalamnya AWS Organizations, tanpa menulis kebijakan berbasis sumber daya individual untuk setiap sumber daya bersama. Meskipun Anda dapat memberikan akses lintas akun dengan melampirkan kebijakan berbasis sumber daya secara langsung ke sumber daya, AWS RAM menyediakan alternatif terpusat dan terkelola yang menghilangkan kebutuhan untuk menghitung ID akun dalam kebijakan atau mempertahankan dokumen kebijakan yang identik di seluruh akun.

Dengan RAM, akun yang dikonsumsi melihat sumber daya bersama secara asli di konsol layanan mereka, pemilik sumber daya mempertahankan kepemilikan penuh dan visibilitas ke siapa yang memiliki akses, dan izin terkelola menentukan tindakan maksimum yang dapat dilakukan konsumen —semua diatur oleh pembagian sumber daya tunggal daripada kebijakan per sumber daya. Kasus penggunaan umum termasuk berbagi subnet VPC, lampiran AWS Transit Gateway, aturan Route 53 Resolver, dan konfigurasi Manajer Lisensi di seluruh akun. Untuk informasi selengkapnya, lihat Apa itu AWS Resource Access Manager? dalam Panduan Pengguna AWS RAM.

Kebijakan sesi

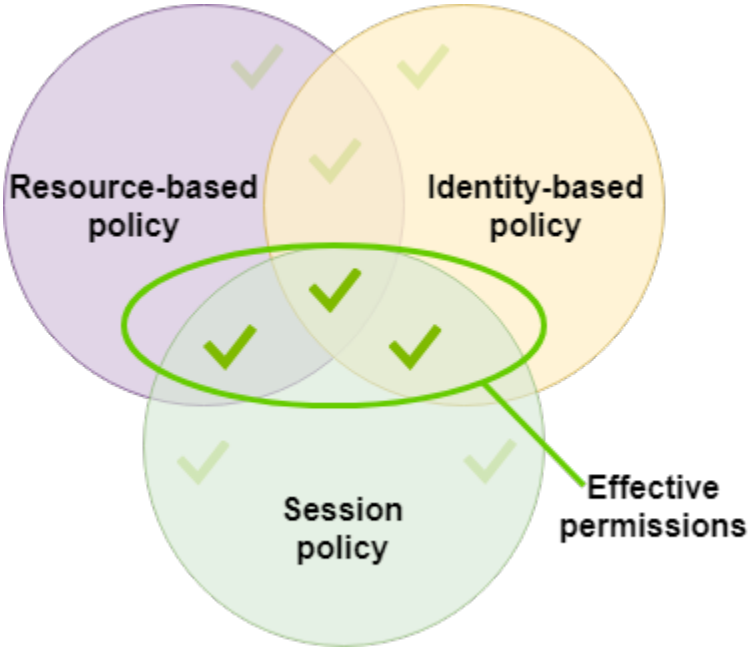
Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau prinsipal pengguna AWS STS federasi. Izin untuk sesi adalah titik temu kebijakan berbasis identitas untuk entitas IAM (pengguna atau peran) yang digunakan untuk membuat sesi dan kebijakan sesi. Izin juga dapat berasal dari kebijakan berbasis sumber daya. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin.

Anda dapat membuat sesi peran dan memberikan kebijakan sesi secara terprogram menggunakan operasi API `AssumeRole`, `AssumeRoleWithSAML`, atau `AssumeRoleWithWebIdentity`. Anda dapat meneruskan suatu dokumen kebijakan sesi inline JSON tunggal dengan menggunakan parameter `Policy`. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Untuk informasi selengkapnya tentang cara membuat sesi peran, lihat Izin untuk kredensial keamanan sementara.

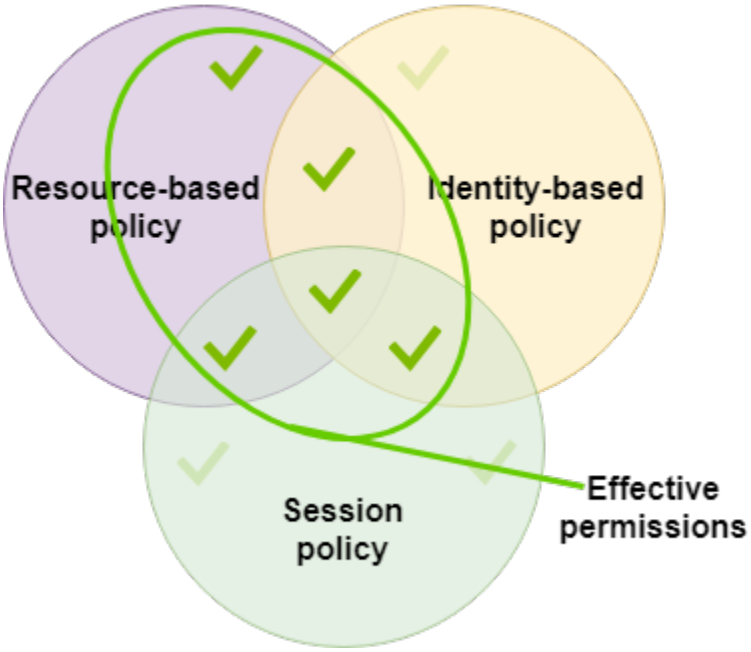
Saat Anda membuat sesi utama AWS STS pengguna federasi, Anda menggunakan kunci akses pengguna IAM untuk memanggil operasi API secara terprogram. `GetFederationToken` Anda juga harus memberikan kebijakan sesi. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis

identitas dan kebijakan sesi. Untuk informasi selengkapnya tentang cara membuat sesi pengguna gabungan, lihat Meminta kredenSIAL melalui broker identitas khusus.

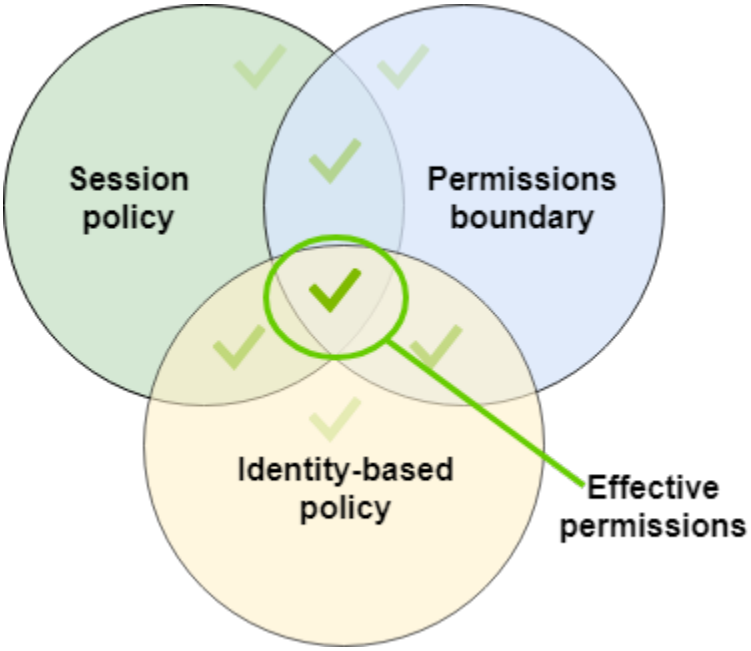
Kebijakan berbasis sumber daya dapat menentukan ARN pengguna atau peran sebagai prinsipal. Dalam hal ini, izin dari kebijakan berbasis sumber daya ditambahkan ke peran atau kebijakan berbasis identitas pengguna sebelum sesi dibuat. Kebijakan sesi membatasi total izin yang diberikan oleh kebijakan berbasis sumber daya dan kebijakan berbasis identitas. Izin sesi yang dihasilkan adalah persimpangan kebijakan sesi dan kebijakan berbasis sumber daya ditambah persimpangan kebijakan sesi dan kebijakan berbasis identitas.



Kebijakan berbasis sumber daya dapat menentukan ARN sesi sebagai prinsipal. Dalam hal ini, izin dari kebijakan berbasis sumber daya ditambahkan setelah sesi dibuat. Izin kebijakan berbasis sumber daya tidak terbatas oleh kebijakan sesi. Izin sesi yang dihasilkan mempunyai semua izin dari kebijakan berbasis sumber daya ditambah titik pertemuan antara kebijakan berbasis identitas dan kebijakan sesi.



Batas izin dapat mengatur izin maksimum untuk pengguna atau peran yang digunakan untuk membuat sebuah sesi. Dalam kasus ini, Izin sesi yang dihasilkan adalah titik pertemuan antara kebijakan kebijakan sesi, batas izin, dan kebijakan berbasis identitas. Namun, batas izin tidak membatasi izin yang diberikan oleh kebijakan berbasis sumber daya yang menentukan ARN dari sesi yang dihasilkan.



Kebijakan dan pengguna root

Ini Pengguna root akun AWS dipengaruhi oleh beberapa jenis kebijakan tetapi tidak yang lain. Anda tidak dapat melampirkan kebijakan berbasis identitas ke pengguna root, dan Anda tidak dapat menetapkan batas izin bagi pengguna root. Namun, Anda dapat menentukan pengguna root sebagai prinsipal dalam kebijakan berbasis sumber daya atau ACL. Pengguna root masih menjadi anggota akun. Jika akun itu adalah anggota organisasi di AWS Organizations, pengguna root dipengaruhi oleh SCP dan RCP untuk akun tersebut.

Gambaran dari kebijakan JSON

Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Identity-based kebijakan dan kebijakan yang digunakan untuk menetapkan batas izin adalah dokumen kebijakan JSON yang Anda lampirkan ke pengguna atau peran. Resource-based kebijakan adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. SCP dan RCP adalah dokumen kebijakan JSON dengan sintaks terbatas yang Anda lampirkan ke AWS Organizations'root organisasi, unit organisasi (OU), atau akun. ACL juga terlampir pada sumber daya, tetapi Anda harus menggunakan sintaks yang berbeda. Kebijakan sesi adalah kebijakan JSON yang berikan saat Anda mengambil sebuah peran atau sesi pengguna gabungan.

Tidak perlu bagi Anda untuk memahami sintaks JSON. Anda dapat menggunakan editor visual di Konsol Manajemen AWS untuk membuat dan mengedit kebijakan yang dikelola pelanggan tanpa pernah menggunakan JSON. Namun, jika Anda menggunakan kebijakan inline untuk grup atau kebijakan yang kompleks, Anda masih harus membuat dan mengubah kebijakan tersebut di editor JSON menggunakan konsol. Untuk informasi lebih lanjut tentang cara menggunakan editor visual, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan dan Edit kebijakan IAM.

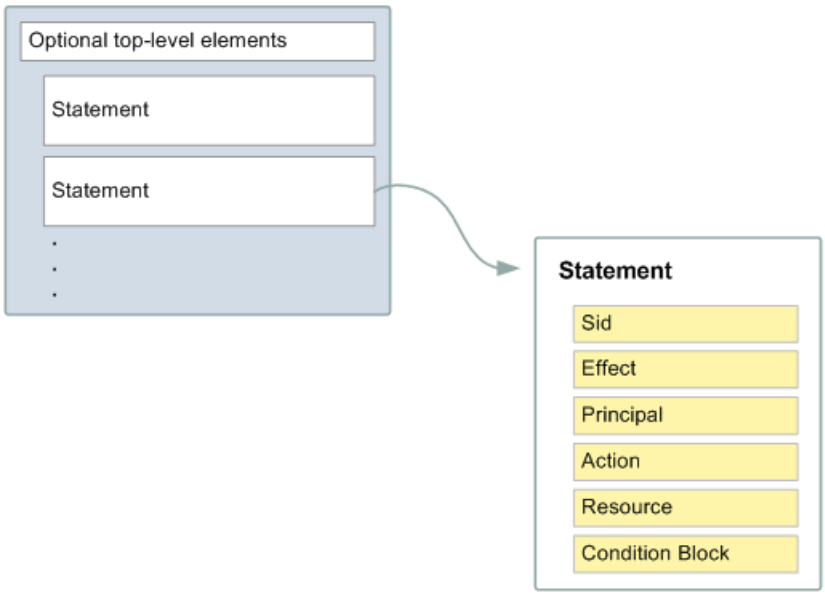
Saat Anda membuat atau mengedit kebijakan JSON, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari lebih lanjut tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Struktur dokumen kebijakan JSON

Sebagaimana digambarkan dalam gambar berikut, dokumen kebijakan JSON mencakup elemen-elemen ini:

- Informasi opsional untuk seluruh kebijakan di bagian atas dokumen
- Satu atau lebih pernyataan individu

Setiap pernyataan mencakup informasi tentang satu izin. Jika kebijakan menyertakan beberapa pernyataan, terap AWS kan logika `OR` di seluruh pernyataan saat mengevaluasinya. Jika beberapa kebijakan AWS berlaku untuk permintaan, terapkan logika `OR` di semua kebijakan tersebut saat mengevaluasinya.



Informasi dalam pernyataan terkandung di dalam rangkaian elemen.

- Version – Tentukan versi bahasa kebijakan yang ingin Anda gunakan. Kami menyarankan Anda menggunakan `2012-10-17` versi terbaru. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM JSON: Version
- Statement – Gunakan elemen kebijakan utama ini sebagai wadah untuk elemen berikut. Anda dapat menyertakan lebih dari satu pernyataan dalam kebijakan.
- Sid (Opsional) – Sertakan ID pernyataan opsional untuk membedakan antara pernyataan Anda.
- Effect – Gunakan `Allow` atau `Deny` untuk menunjukkan apakah kebijakan mengizinkan atau menolak akses.
- Principal(Diperlukan dalam beberapa keadaan) — Jika Anda membuat kebijakan berbasis sumber daya, Anda harus menunjukkan akun, pengguna, peran, atau prinsipal pengguna AWS STS federasi yang ingin Anda izinkan atau menolak aksesnya. Jika Anda membuat kebijakan izin IAM

untuk melampirkan ke pengguna atau peran, Anda tidak dapat menyertakan elemen ini. Prinsipal tersirat sebagai pengguna atau peran tersebut.

- Action – Sertakan daftar tindakan yang diperbolehkan atau ditolak oleh kebijakan.
- Resource(Diperlukan dalam beberapa keadaan) — Jika Anda membuat kebijakan izin IAM, Anda harus menentukan daftar sumber daya yang diterapkan tindakan tersebut. Jika Anda membuat kebijakan berbasis sumber daya, itu tergantung pada sumber daya yang Anda gunakan untuk mengetahui apakah elemen ini diperlukan atau tidak.
- Condition (Opsional) – Tentukan keadaan di mana kebijakan memberikan izin.

Untuk mempelajari tentang ini dan elemen kebijakan yang lebih maju lainnya, lihat Referensi elemen kebijakan IAM JSON.

Beberapa pernyataan dan beberapa kebijakan

Jika Anda ingin menetapkan lebih dari satu izin untuk suatu entitas (pengguna atau peran), Anda dapat menggunakan beberapa pernyataan dalam satu kebijakan. Anda juga dapat melampirkan beberapa kebijakan. Jika Anda mencoba untuk menentukan beberapa izin dalam satu pernyataan, kebijakan Anda mungkin tidak memberikan akses yang Anda harapkan. Sebaiknya bagi kebijakan berdasarkan jenis sumber daya.

Karena ukuran kebijakan yang terbatas, mungkin perlu menggunakan beberapa kebijakan untuk izin yang lebih kompleks. Ini juga merupakan ide bagus untuk membuat pengelompokan fungsional izin dalam kebijakan pengelolaan pelanggan yang terpisah. Misalnya, Buat satu kebijakan untuk manajemen pengguna IAM, satu untuk manajemen mandiri, dan kebijakan lain untuk manajemen bucket S3. Terlepas dari kombinasi beberapa pernyataan dan beberapa kebijakan, evaluasi AWS kebijakan Anda dengan cara yang sama.

Misalnya, kebijakan berikut memiliki tiga pernyataan, yang masing-masing mendefinisikan satu set izin secara terpisah dalam satu akun. Pernyataan tersebut menjelaskan hal berikut:

- Pernyataan pertama, dengan sebuah `Sid` (ID Pernyataan) dari `FirstStatement`, memungkinkan pengguna dengan kebijakan terlampir untuk mengubah kata sandi milik mereka. Elemen `Resource` dalam pernyataan ini adalah “*“ (yang berarti “semua sumber daya”). Namun dalam pelaksanaannya, Operasi API `ChangePassword` (atau setara Perintah CLI `change-password`) hanya memengaruhi kata sandi untuk pengguna yang membuat permintaan.
- Pernyataan kedua memungkinkan pengguna mencantumkan semua bucket Amazon S3 di dalamnya Akun AWS. Elemen `Resource` dalam pernyataan ini adalah `"*"` (yang berarti “semua

sumber daya"). Tetapi karena kebijakan tidak memberikan akses ke sumber daya di akun lain, pengguna hanya dapat mencantumkan bucket di akun mereka sendiri Akun AWS.

- Pernyataan ketiga memungkinkan pengguna membuat daftar dan mengambil objek apa saja yang ada di dalam bucket bernama `amzn-s3-demo-bucket-confidential-data`, tetapi hanya saat pengguna diautentikasi dengan Autentikasi Multi-Faktor (MFA). Elemen `Condition` dalam kebijakan tersebut memberlakukan autentikasi MFA.

  Saat pernyataan kebijakan memuat elemen `Condition`, pernyataan hanya berlaku saat elemen `Condition` dievaluasi menjadi benar. Dalam hal ini, `Condition` evaluasi menjadi true ketika pengguna berada MFA-authenticated. Jika pengguna tidak MFA-authenticated, ini di `Condition` evaluasi menjadi false. Dalam hal ini, pernyataan ketiga dalam kebijakan ini tidak berlaku dan pengguna tidak memiliki akses ke bucket `amzn-s3-demo-bucket-confidential-data`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data",
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data/*"
```

```
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Contoh sintaks kebijakan JSON

Kebijakan berbasis identitas berikut mengizinkan prinsipal tersirat untuk mencantumkan satu bucket Amazon S3 dengan nama `amzn-s3-demo-bucket`:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

Kebijakan berbasis sumber daya berikut dapat dilampirkan ke bucket Amazon S3 Kebijakan ini memungkinkan anggota tertentu Akun AWS untuk melakukan tindakan Amazon S3 apa pun di bucket yang diberi nama`amzn-s3-demo-bucket`. Hal ini mengizinkan tindakan apa pun yang dapat dilakukan pada bucket atau objek di dalamnya. (Karena kebijakan tersebut hanya memberikan kepercayaan ke akun, pengguna individu dalam akun tersebut harus tetap diberikan izin untuk tindakan Amazon S3 yang ditentukan.)

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
```

```
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Untuk melihat contoh kebijakan untuk skenario umum, lihat Contoh kebijakan berbasis identitas IAM.

Berikan hak akses paling rendah

Saat Anda membuat kebijakan IAM, ikuti saran keamanan standar untuk memberikan hak istimewa paling rendah, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Tentukan apa yang perlu dilakukan pengguna dan peran dan kemudian buat kebijakan yang memungkinkan mereka melakukan hanya tugas-tugas tersebut.

Mulai dengan satu set izin minimum dan berikan izin tambahan sesuai kebutuhan. Melakukan hal tersebut lebih aman daripada memulai dengan izin yang terlalu fleksibel, lalu mencoba memperketatnya nanti.

Sebagai alternatif dari hak istimewa terkecil, Anda dapat menggunakan AWS kebijakan ter kelola atau kebijakan dengan * izin wildcard untuk memulai kebijakan. Pertimbangkan risiko keamanan dengan memberi kepala sekolah Anda lebih banyak izin daripada yang mereka butuhkan untuk melakukan pekerjaan mereka. Pantau prinsipal tersebut untuk mempelajari izin mana yang mereka gunakan. Kemudian tulis kebijakan hak istimewa paling sedikit.

IAM menyediakan beberapa opsi untuk membantu Anda memperbaiki izin yang Anda berikan.

- Memahami pengelompokan tingkat akses — Anda dapat menggunakan pengelompokan tingkat akses untuk memahami tingkat akses yang diberikan kebijakan. Tindakan kebijakan diklasifikasikan sebagai `List`, `Read`, `Write`, `Permissions management`, atau `Tagging`. Misalnya, Anda dapat memilih tindakan dari tingkat akses `List` dan `Read` untuk memberikan akses hanya-baca kepada pengguna Anda. Untuk mempelajari cara menggunakan ringkasan kebijakan untuk memahami izin tingkat akses, lihat Tingkat akses dalam ringkasan kebijakan.

- Memvalidasi kebijakan Anda — Anda dapat melakukan validasi kebijakan menggunakan IAM Access Analyzer saat membuat dan mengedit kebijakan JSON. Sebaiknya Anda meninjau dan memvalidasi semua kebijakan yang ada. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan untuk memvalidasi kebijakan Anda. Ini menghasilkan peringatan keamanan ketika pernyataan dalam kebijakan Anda memungkinkan akses yang kita anggap terlalu permisif. Anda dapat menggunakan rekomendasi yang dapat ditindaklanjuti yang disediakan melalui peringatan keamanan saat Anda berupaya memberikan hak istimewa paling rendah. Untuk mempelajari lebih lanjut tentang pemeriksaan kebijakan yang disediakan oleh IAM Access Analyzer, lihat validasi kebijakan IAM Access Analyzer.
- Buat kebijakan berdasarkan aktivitas akses — Untuk membantu Anda menyempurnakan izin yang Anda berikan, Anda dapat membuat kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas IAM (pengguna atau peran). IAM Access Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang telah digunakan oleh entitas dalam kerangka waktu yang Anda tentukan. Anda dapat menggunakan template untuk membuat kebijakan terkelola dengan izin berbutir halus dan kemudian melampirkannya ke entitas IAM. Dengan begitu, Anda hanya memberikan izin yang dibutuhkan pengguna atau peran untuk berinteraksi dengan AWS sumber daya untuk kasus penggunaan spesifik Anda. Untuk mempelajari selengkapnya, lihat pembuatan kebijakan Penganalisis Akses IAM.
- Gunakan informasi yang terakhir diakses — Fitur lain yang dapat membantu dengan hak istimewa paling sedikit adalah informasi yang terakhir diakses. Lihat informasi ini di tab Penasihat Akses pada halaman detail konsol IAM untuk pengguna, grup, peran, atau kebijakan IAM. Informasi terakhir yang diakses juga mencakup informasi tentang tindakan yang terakhir diakses untuk beberapa layanan, seperti Amazon EC2, IAM, Lambda, dan Amazon S3. Jika Anda masuk menggunakan kredentif akun AWS Organizations manajemen, Anda dapat melihat informasi layanan yang terakhir diakses di AWS Organizations bagian konsol IAM. Anda juga dapat menggunakan AWS API AWS CLI atau untuk mengambil laporan informasi terakhir yang diakses untuk entitas atau kebijakan di IAM atau AWS Organizations. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan IAM atau AWS Organizations kebijakan agar lebih mematuhi prinsip hak istimewa terendah. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.
- Tinjau peristiwa akun di AWS CloudTrail — Untuk mengurangi izin lebih lanjut, Anda dapat melihat peristiwa akun Anda di Ri AWS CloudTrail wayat acara. CloudTrail log peristiwa menyertakan informasi peristiwa terperinci yang dapat Anda gunakan untuk mengurangi izin kebijakan. Log hanya mencakup tindakan dan sumber daya yang diperlukan oleh entitas IAM Anda. Untuk informasi selengkapnya, lihat CloudTrail Melihat Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.

Untuk informasi selengkapnya, lihat topik kebijakan berikut untuk layanan individual, yang memberikan contoh cara menulis kebijakan untuk sumber daya khusus layanan.

- Menggunakan kebijakan berbasis sumber daya untuk DynamoDB di Panduan Pengembang Amazon Dynam o DB
- Kebijakan bucket untuk Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- Ikhtisar Daftar Kontrol Akses (ACL) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon

Kebijakan terkelola dan kebijakan inline

Saat menetapkan izin untuk identitas di IAM, Anda harus memutuskan apakah akan menggunakan kebijakan terkel AWS ola, kebijakan yang dikelola pelanggan, atau kebijakan sebaris. Topik berikut memberikan informasi selengkapnya tentang masing-masing jenis kebijakan berbasis identitas dan kapan menggunakannya.

Tabel berikut menguraikan kebijakan ini:

Jenis Kebijakan	Deskripsi	Siapa yang mengelola kebijakan tersebut?	Ubah izin?	Jumlah prinsipal yang diterapkan pada kebijakan?
AWS kebijakan terkelola	Kebijakan mandiri dibuat dan dikelola oleh AWS.	AWS	Tidak	Banyak
Kebijakan yang dikelola pelanggan	Kebijakan yang Anda buat untuk kasus penggunaan tertentu, dan Anda dapat mengubah atau memperbaruinya	Anda	Ya	Banyak

Jenis Kebijakan	Deskripsi	Siapa yang mengelola kebijakan tersebut?	Ubah izin?	Jumlah prinsipal yang diterapkan pada kebijakan?
	sesering yang Anda inginkan.			
Kebijakan inline	Kebijakan dibuat untuk identitas IAM tunggal (pengguna, grup, atau peran) yang mempertahankan hubungan satu-ke-satu yang ketat antara kebijakan dan identitas.	Anda	Ya	Satu

Untuk petunjuk tentang membuat peran dan melampirkan kebijakan ini, lihatMenciptakan peran dan memberlakukan kebijakan (konsol).

Topik

- AWS kebijakan terkelola
- Kebijakan yang dikelola pelanggan
- Kebijakan inline
- Pilih antara kebijakan terkelola dan kebijakan inline
- Mengonversi kebijakan inline menjadi kebijakan terkelola
- Usang AWS kebijakan terkelola

AWS kebijakan terkelola

Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan mandiri berarti kebijakan memiliki Nama Sumber Daya Amazon (ARN) sendiri yang menyertakan

nama kebijakan. Misalnya, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` adalah kebijakan yang AWS dikelola. Untuk informasi selengkapnya tentang ARN, lihat ARN IAM Untuk daftar kebijakan terkel AWS ola untuk Layanan AWS, lihat kebijakan ter AWS kelola.

AWS kebijakan terkelola memudahkan Anda untuk menetapkan izin yang sesuai untuk pengguna, grup IAM, dan peran. Ini lebih cepat daripada menulis kebijakan sendiri, dan menyertakan izin untuk banyak kasus penggunaan umum.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. AWS sesekali memperbarui izin yang ditentukan dalam kebijakan AWS terkelola. Ketika AWS melakukan ini, pembaruan memengaruhi semua entitas utama (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan. AWS kemungkinan besar akan memperbarui kebijakan ter AWS kelola saat AWS layanan baru diluncurkan atau panggilan API baru tersedia untuk layanan yang ada. Misalnya, kebijakan terkel AWS ola yang disebut ReadOnlyAccess menyediakan akses baca-saja ke semua Layanan AWS dan sumber daya. Saat AWS meluncurkan layanan baru, per AWS barui ReadOnlyAccess kebijakan untuk menambahkan izin baca-saja untuk layanan baru. Izin yang diperbarui diterapkan pada semua entitas prinsipal yang kebijakannya dilampirkan.

Kebijakan ter AWS kelola akses penuh: Kebijakan ini menentukan izin untuk administrator layanan dengan memberikan akses penuh ke layanan. Contohnya termasuk:

- AmazonDynamoDBFullAccess
- IAMFullAccess

Power-user AWS kebijakan terkelola: Kebijakan ini menyediakan akses penuh ke Layanan AWS dan sumber daya, tetapi tidak mengizinkan pengelolaan pengguna dan grup IAM. Contohnya termasuk:

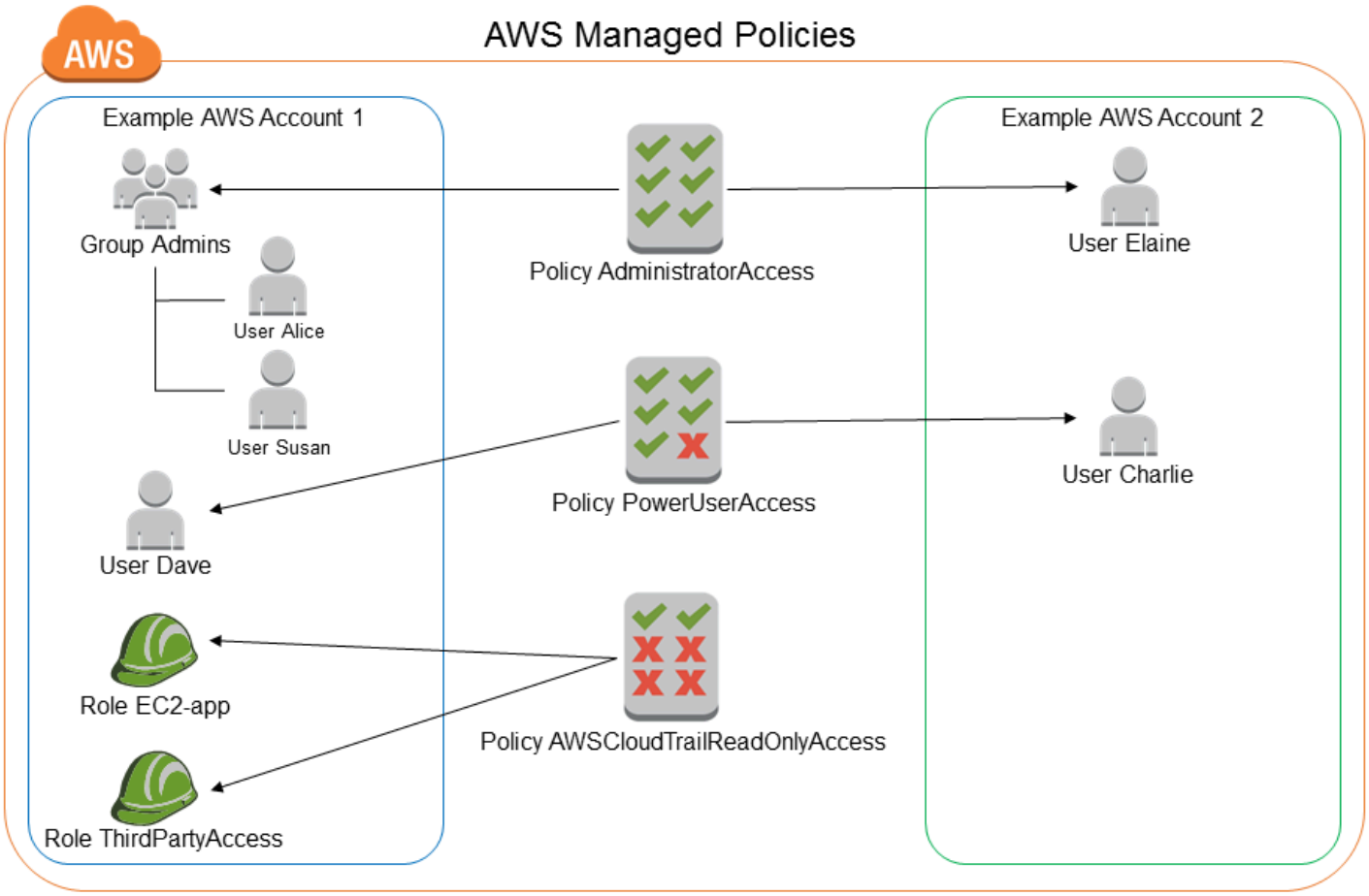
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

Partial-access AWS kebijakan ter kelola: Ini menyediakan tingkat akses tertentu Layanan AWS tanpa mengizinkan izin tingkat akses manajemen izin. Contohnya termasuk:

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

Kebijakan yang AWS dikelola fungsi pekerjaan: Kebijakan ini selaras dengan fungsi pekerjaan yang umum digunakan di industri TI dan memfasilitasi pemberian izin untuk fungsi pekerjaan ini. Salah satu keuntungan utama menggunakan kebijakan fungsi pekerjaan adalah bahwa kebijakan tersebut dipelihara dan diperbarui AWS saat layanan baru dan operasi API diperkenalkan. Misalnya, fungsi AdministratorAccess pekerjaan menyediakan akses penuh dan delegasi izin ke setiap layanan dan sumber daya di AWS. Sebaiknya gunakan kebijakan ini hanya untuk administrator akun. Untuk pengguna canggih yang memerlukan akses penuh ke setiap layanan kecuali akses terbatas ke IAM dan AWS Organizations, gunakan fungsi PowerUserAccess pekerjaan. Untuk daftar dan deskripsi kebijakan fungsi pekerjaan, lihat AWS kebijakan terkelola untuk fungsi pekerjaan.

Diagram berikut menggambarkan kebijakan AWS terkelola. Diagram menunjukkan tiga kebijakan AWS terkelola: AdministratorAccess PowerUserAccess,, dan AWS CloudTrail _ ReadOnlyAccess. Perhatikan bahwa kebijakan terkel AWS ola tunggal dapat dilampirkan ke entitas utama yang berbeda Akun AWS, dan ke entitas utama yang berbeda dalam satu entitas Akun AWS.



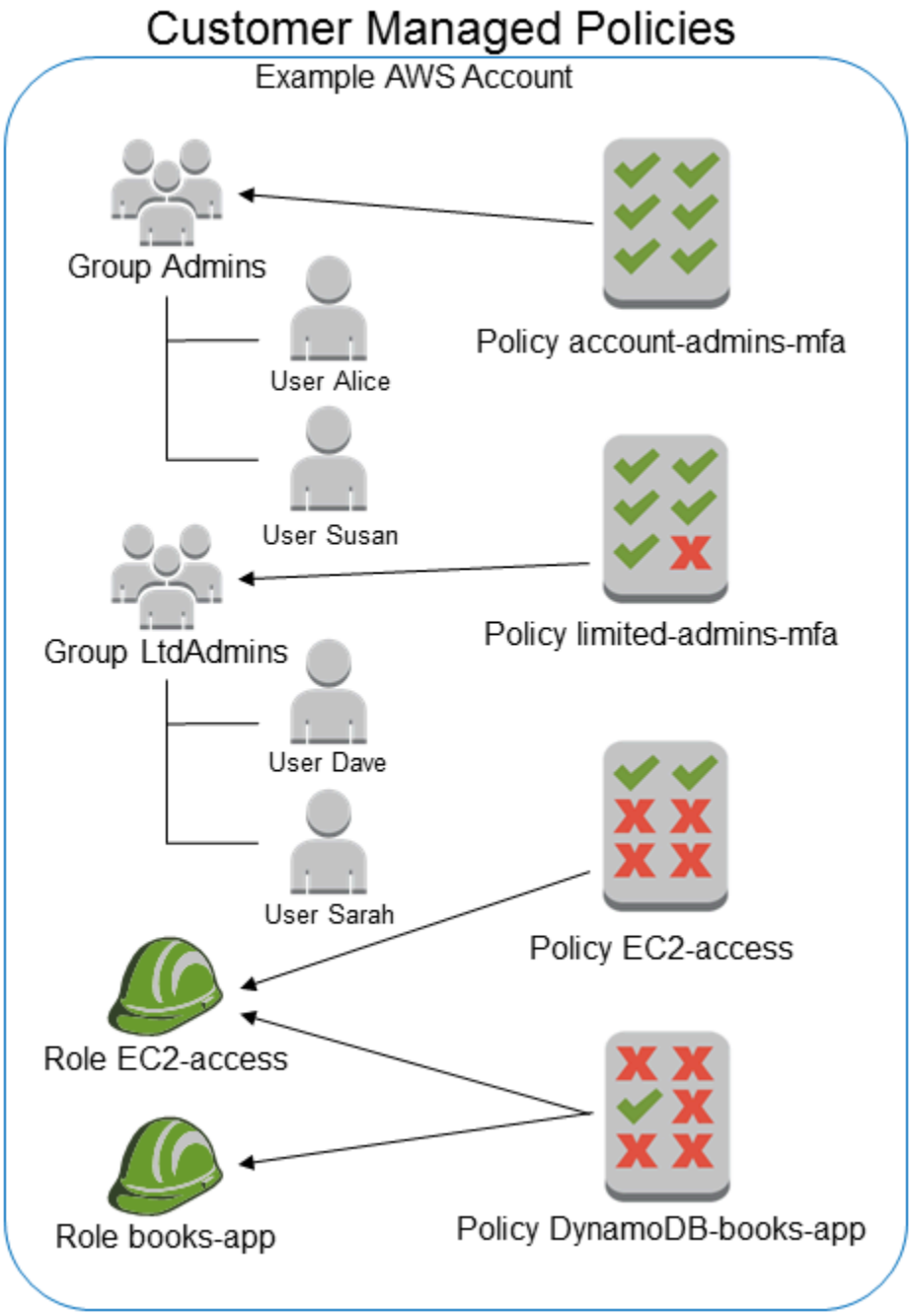
Kebijakan yang dikelola pelanggan

Anda dapat membuat kebijakan mandiri sendiri Akun AWS yang dapat Anda lampirkan ke entitas utama (pengguna IAM, grup IAM, dan peran IAM). Anda membuat kebijakan yang dikelola pelanggan ini untuk kasus penggunaan spesifik Anda, dan Anda dapat mengubah dan memperbaruinya sesering yang Anda inginkan. Seperti AWS kebijakan terkelola, saat Anda melampirkan kebijakan ke entitas utama, Anda memberi entitas izin yang ditentukan dalam kebijakan. Saat Anda memperbarui izin dalam kebijakan, perubahan diterapkan ke semua entitas utama yang dilampirkan kebijakan tersebut.

Cara terbaik untuk membuat kebijakan yang dikelola pelanggan adalah memulai dengan menyalin kebijakan terkelola AWS yang sudah ada. Dengan demikian Anda tahu bahwa kebijakan tersebut benar sejak awal dan yang perlu Anda lakukan hanyalah menyesuaikan itu dengan lingkungan Anda.

Diagram berikut menggambarkan kebijakan yang dikelola pelanggan. Setiap kebijakan adalah entitas di IAM dengan Amazon Resource Name (ARN) miliknya sendiri yang termasuk nama kebijakan. Perhatikan bahwa kebijakan yang sama dapat dilampirkan ke beberapa entitas utama—misalnya, DynamoDB-books-app kebijakan yang sama dilampirkan ke dua peran IAM yang berbeda.

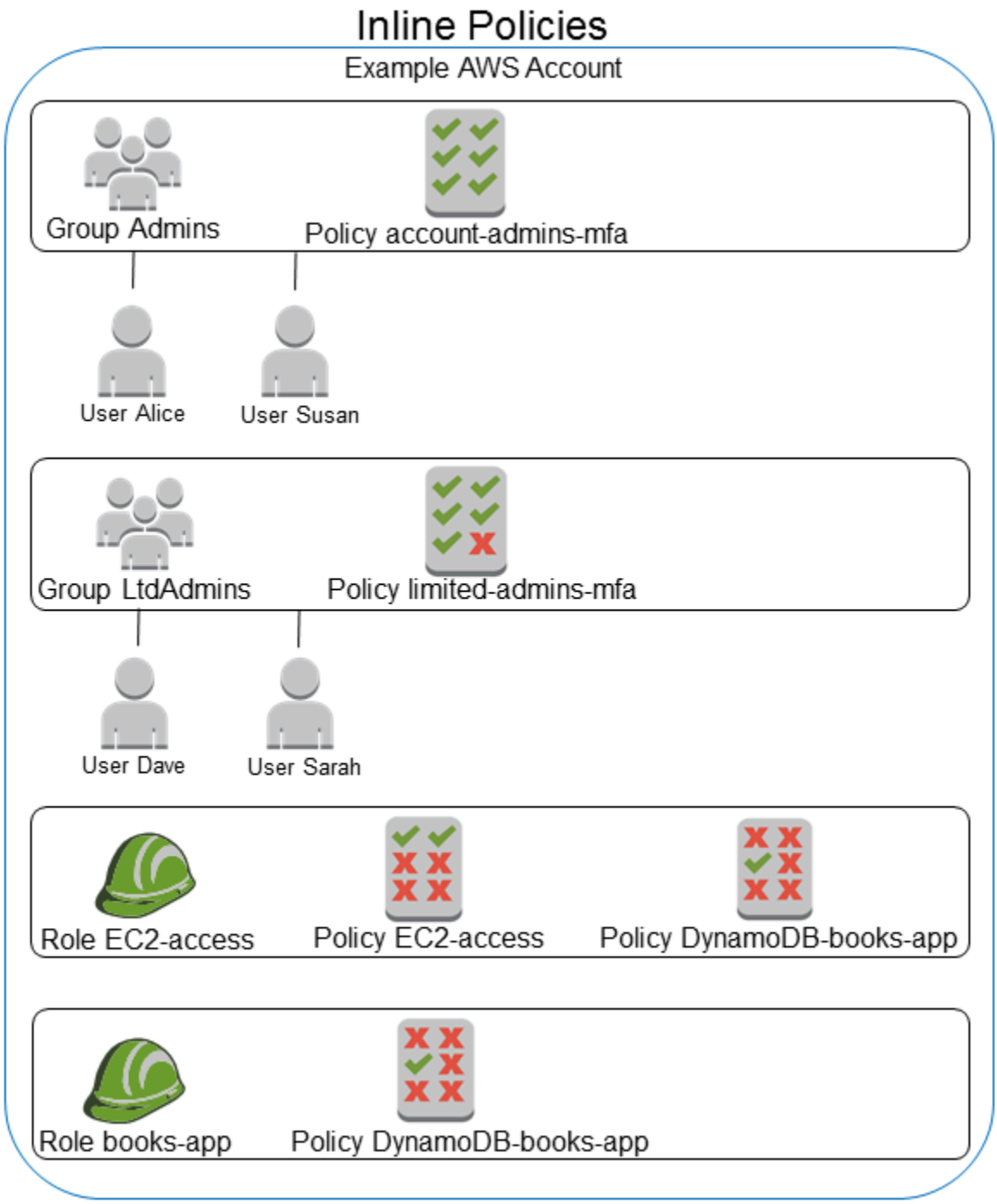
Untuk informasi selengkapnya, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan



Kebijakan inline

Kebijakan sebaris adalah kebijakan yang dibuat untuk identitas IAM tunggal (pengguna, grup pengguna, atau peran). Kebijakan inline mempertahankan hubungan satu-ke-satu yang ketat antara kebijakan dan sebuah identitas. Mereka dihapus ketika Anda menghapus identitas. Anda dapat membuat kebijakan dan menyematkannya dalam identitas, baik saat Anda membuat identitas atau nanti. Jika kebijakan dapat diterapkan ke lebih dari satu entitas, sebaiknya gunakan kebijakan terkelola.

Diagram berikut menggambarkan kebijakan inline. Setiap kebijakan merupakan bagian yang melekat pada pengguna, grup, atau peran. Perhatikan bahwa dua peran menyertakan kebijakan yang sama (DynamoDB-books-appkebijakan), tetapi mereka tidak berbagi kebijakan tunggal. Setiap peran memiliki salinan kebijakannya sendiri.



Pilih antara kebijakan terkelola dan kebijakan inline

Pertimbangkan kasus penggunaan Anda saat memutuskan antara kebijakan terkelola dan sebaris. Dalam sebagian besar kasus, kami menyarankan agar Anda menggunakan kebijakan terkelola daripada kebijakan inline.

> **Note**
>
> Anda dapat menggunakan kebijakan terkelola dan sebaris bersama-sama untuk menentukan izin umum dan unik untuk entitas utama.

Kebijakan terkelola menyediakan fitur berikut:

Dapat digunakan kembali

Satu kebijakan terkelola dapat dilampirkan pada beberapa entitas prinsipal (pengguna, grup, dan peran). Anda dapat membuat pustaka kebijakan yang menentukan izin berguna untuk Anda Akun AWS, lalu melampirkan kebijakan ini ke entitas utama sesuai kebutuhan.

Manajemen perubahan pusat

Ketika Anda mengubah kebijakan terkelola, perubahan diterapkan pada semua entitas prinsipal yang terkait dengan kebijakan tersebut. Misalnya, jika ingin menambahkan izin untuk AWS API baru, Anda dapat memperbarui kebijakan yang dikelola pelanggan atau mengaitkan kebijakan AWS terkelola untuk menambahkan izin. Jika Anda menggunakan kebijakan AWS terkelola, AWS perbarui kebijakan tersebut. Ketika kebijakan terkelola diperbarui, perubahan diterapkan ke semua entitas utama yang dilampirkan pada kebijakan terkelola. Sebaliknya, untuk mengubah kebijakan inline, Anda harus mengedit secara individual setiap identitas yang berisi kebijakan inline. Misalnya, jika grup dan peran keduanya berisi kebijakan inline yang sama, Anda harus mengedit kedua entitas utama secara individual untuk mengubah kebijakan tersebut.

Versioning dan peluncuran kembali

Ketika Anda mengubah kebijakan yang dikelola pelanggan, perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan terkelola. IAM menyimpan hingga lima versi kebijakan terkelola pelanggan Anda. Anda dapat menggunakan versi kebijakan untuk mengembalikan kebijakan ke versi sebelumnya sesuai kebutuhan.

> **Note**
>
> Versi kebijakan berbeda dari elemen kebijakan `Version`. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Untuk mempelajari lebih lanjut tentang versi kebijakan, lihat the section called “Peningkatan versi IAM”. Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat Elemen kebijakan IAM JSON: Version.

Menyerahkan manajemen perizinan

Anda dapat mengizinkan pengguna Akun AWS untuk melampirkan dan melepaskan kebijakan sambil mempertahankan kendali atas izin yang ditentukan dalam kebijakan tersebut. Untuk melakukan ini, tetapkan beberapa pengguna sebagai administrator lengkap—yaitu administrator yang dapat membuat, memperbarui, dan menghapus kebijakan. Kemudian Anda dapat menunjuk pengguna lain sebagai administrator terbatas. Administrator terbatas tersebut dapat melampirkan kebijakan ke entitas utama lainnya, tetapi hanya kebijakan yang Anda izinkan untuk dilampirkan.

Untuk informasi lebih lanjut tentang menyerahkan manajemen perizinan, lihat Mengendalikan akses ke kebijakan.

Batas karakter kebijakan yang lebih besar

Batas ukuran karakter maksimum untuk kebijakan terkelola lebih besar dari batas karakter untuk kebijakan sebaris grup. Jika Anda mencapai batas ukuran karakter kebijakan sebaris, Anda dapat membuat grup IAM lainnya dan melampirkan kebijakan terkelola ke grup.

Untuk informasi lebih lanjut tentang kuota dan batas, lihatIAM dan AWS STS kuota.

Pembaruan otomatis untuk kebijakan AWS terkelola

AWS mempertahankan kebijakan AWS terkelola dan memperbaruinya bila perlu, misalnya, untuk menambahkan izin untuk AWS layanan baru, tanpa Anda harus membuat perubahan. Pembaruan secara otomatis diterapkan ke entitas utama tempat Anda melampirkan kebijakan AWS terkelola.

Memulai kebijakan terkelola

Sebaiknya gunakan kebijakan yang memberikan hak istimewa paling sedikit, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Cara paling aman untuk memberikan hak istimewa paling sedikit adalah dengan menulis kebijakan yang dikelola pelanggan hanya dengan izin yang diperlukan oleh tim Anda. Anda harus membuat proses untuk memungkinkan tim Anda meminta lebih

banyak izin bila diperlukan. Dibutuhkan waktu dan keahlian untuk membuat kebijakan yang dikelola pelanggan IAM yang hanya memberi tim Anda izin yang mereka butuhkan.

Untuk mulai menambahkan izin ke identitas IAM Anda (pengguna, grup pengguna, dan peran), Anda dapat menggunakannya. AWS kebijakan terkelola AWS kebijakan terkelola tidak memberikan izin hak istimewa paling sedikit. Anda harus mempertimbangkan risiko keamanan memberikan izin kepada kepala sekolah Anda lebih banyak daripada yang mereka butuhkan untuk melakukan pekerjaan mereka.

Anda dapat melampirkan kebijakan AWS terkelola, termasuk fungsi pekerjaan, ke identitas IAM apa pun. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Untuk beralih ke izin hak istimewa terkecil, Anda dapat menjalankan AWS Identity and Access Management dan Access Analyzer untuk memantau prinsipal dengan kebijakan terkelola. AWS Setelah mengetahui izin yang mereka gunakan, Anda dapat menulis atau membuat kebijakan yang dikelola pelanggan hanya dengan izin yang diperlukan untuk tim Anda. Ini kurang aman, tetapi memberikan lebih banyak fleksibilitas saat Anda mempelajari bagaimana tim Anda menggunakan AWS. Untuk informasi selengkapnya, lihat Pembuatan kebijakan Penganalisis Akses IAM.

AWS kebijakan terkelola dirancang untuk memberikan izin untuk banyak kasus penggunaan umum. Untuk informasi selengkapnya tentang kebijakan AWS terkelola yang dirancang untuk fungsi pekerjaan tertentu, lihatAWS kebijakan terkelola untuk fungsi pekerjaan.

Untuk daftar kebijakan AWS terkelola, lihat Panduan Referensi Kebijakan AWS Terkelola.

Menggunakan kebijakan inline

Kebijakan inline berguna jika Anda ingin mempertahankan one-to-one hubungan yang ketat antara kebijakan dan identitas yang diterapkan. Misalnya, jika Anda ingin memastikan bahwa izin dalam kebijakan tidak secara tidak sengaja ditetapkan ke identitas selain identitas yang dimaksudkan. Ketika Anda menggunakan kebijakan inline, izin dalam kebijakan tersebut tidak dapat secara tidak sengaja dilampirkan pada identitas yang salah. Selain itu, ketika Anda menggunakan Konsol Manajemen AWS untuk menghapus identitas itu, kebijakan yang disematkan dalam identitas juga dihapus karena mereka adalah bagian dari entitas utama.

Mengonversi kebijakan inline menjadi kebijakan terkelola

Jika Anda memiliki kebijakan inline di akun Anda, Anda dapat mengonversinya ke kebijakan terkelola. Untuk melakukannya, salin kebijakan tersebut ke kebijakan baru yang dikelola. Selanjutnya, lampirkan kebijakan baru ke identitas yang memiliki kebijakan inline. Kemudian hapus kebijakan inline.

Mengonversi kebijakan inline menjadi kebijakan terkelola

Untuk mengonversi kebijakan inline ke kebijakan terkelola

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, atauPeran.
3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Untuk grup IAM, pilih nama kebijakan sebaris yang ingin Anda hapus. Untuk pengguna dan peran, pilih Tampilkan ***n*** lebih banyak, jika perlu, lalu perluas kebijakan sebaris yang ingin Anda hapus.
6. Pilih Salin untuk menyalin dokumen kebijakan JSON untuk kebijakan tersebut.
7. Di panel navigasi, pilih Kebijakan.
8. Pilih Buat kebijakan dan kemudian pilih opsi JSON.
9. Ganti teks yang ada dengan teks kebijakan JSON Anda, lalu pilih Berikutnya.
10. Masukkan nama dan deskripsi opsional untuk kebijakan Anda dan pilih Buat kebijakan.
11. Di panel navigasi, pilih Grup pengguna, Pengguna, atau Peran, lalu pilih kembali nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
12. Pilih tab Izin dan kemudian pilih Tambahkan izin.
13. Untuk grup IAM, pilih kotak centang di samping nama kebijakan baru Anda, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan. Untuk pengguna atau peran, pilih Tambahkan izin. Di halaman berikutnya, pilih Lampirkan kebijakan yang ada secara langsung, pilih kotak centang di sebelah nama kebijakan baru Anda, pilih Berikutnya, lalu pilih Tambahkan izin.

    Anda kembali ke halaman Ringkasan untuk grup pengguna, pengguna, atau peran Anda.
14. Pilih kotak centang di samping kebijakan sebaris yang ingin Anda hapus dan pilih Hapus.

Usang AWS kebijakan terkelola

Untuk menyederhanakan penetapan izin, AWS berikan kebijakan terkel ola — kebijakan yang telah ditentukan sebelumnya yang siap dilampirkan ke pengguna, grup, dan peran IAM Anda.

Terkadang AWS perlu menambahkan izin baru ke kebijakan yang ada, seperti saat layanan baru diperkenalkan. Menambahkan izin baru ke kebijakan yang sudah ada tidak akan mengganggu atau menghilangkan fitur atau kemampuan.

Namun, AWS mungkin memilih untuk membuat kebijakan baru ketika perubahan yang diperlukan dapat berdampak pada pelanggan jika diterapkan pada kebijakan yang ada. Misalnya, menghapus izin dari kebijakan yang sudah ada dapat merusak izin dari suatu entitas atau aplikasi IAM yang bergantung di atasnya, yang berpotensi mengganggu operasi yang bersifat kritis.

Oleh karena itu, ketika perubahan seperti itu diperlukan AWS , buat kebijakan yang sama sekali baru dengan perubahan yang diperlukan dan membuatnya tersedia untuk pelanggan. Kebijakan lama selanjutnya ditandai usang. Untuk informasi selengkapnya, lihat Kebijakan ter kelola yang tidak digunakan AWS lagi di Panduan Referensi Kebijakan AWS Terkelola.

Tetapkan pagar pembatas izin dengan menggunakan perimeter data

Pagar pembatas perimeter data dimaksudkan untuk berfungsi sebagai batas yang selalu aktif untuk membantu melindungi data Anda di seluruh rangkaian akun dan sumber daya yang luas. AWS Perimeter data mengikuti praktik terbaik keamanan IAM untuk membuat pagar pembatas izin di beberapa akun. Pagar pembatas izin di seluruh organisasi ini tidak menggantikan kontrol akses berbutir halus Anda yang ada. Sebaliknya, mereka berfungsi sebagai kontrol akses kasar yang membantu meningkatkan strategi keamanan Anda dengan memastikan pengguna, peran, dan sumber daya mematuhi serangkaian standar keamanan yang ditentukan.

Perimeter data adalah kumpulan pagar pembatas izin di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya Anda yang mengakses sumber daya tepercaya dari jaringan yang diharapkan.

- Identitas tepercaya: Prinsipal (peran atau pengguna IAM) di AWS akun dan AWS layanan Anda yang bertindak atas nama Anda.
- Sumber daya tepercaya: Sumber daya yang dimiliki oleh AWS akun Anda atau oleh AWS layanan yang bertindak atas nama Anda.
- Jaringan yang diharapkan: Pusat data lokal dan cloud pribadi virtual (VPC), atau jaringan AWS layanan yang bertindak atas nama Anda.

> **Note**
>
> Dalam beberapa kasus, Anda mungkin perlu memperluas perimeter data Anda untuk juga menyertakan akses oleh mitra bisnis tepercaya Anda. Anda harus mempertimbangkan semua pola akses data yang dimaksudkan saat membuat definisi identitas tepercaya, sumber daya tepercaya, dan jaringan yang diharapkan khusus untuk perusahaan Anda dan penggunaan Anda Layanan AWS.

Kontrol perimeter data harus diperlakukan sebagai kontrol keamanan lainnya dalam program keamanan informasi dan manajemen risiko. Ini berarti Anda harus melakukan analisis ancaman untuk mengidentifikasi potensi risiko dalam lingkungan cloud Anda, dan kemudian, berdasarkan kriteria penerimaan risiko Anda sendiri, pilih dan terapkan kontrol perimeter data yang sesuai. Untuk menginformasikan pendekatan berbasis risiko berulang untuk implementasi perimeter data dengan lebih baik, Anda perlu memahami risiko keamanan dan vektor ancaman apa yang ditangani oleh kontrol perimeter data serta prioritas keamanan Anda.

Kontrol perimeter data

Kontrol berbutir kasar perimeter data membantu Anda mencapai enam tujuan keamanan yang berbeda di tiga perimeter data melalui penerapan kombinasi dan kunci kondisi yang berbeda. Jenis kebijakan ???

Perimeter	Tujuan kontrol	Penggunaan	Diterapkan pada	Kunci konteks kondisi global
Identitas	Hanya identitas tepercaya yang dapat mengakses sumber daya saya	RCP	Sumber daya	aws: Principal Org ID aws: Principal OrgPaths aws: Principal Account aws: Principal IsAwsService
	Hanya identitas tepercaya yang diizinkan dari jaringan saya	Kebijakan titik akhir VPC	Jaringan	

Perimeter	Tujuan kontrol	Penggunaan	Diterapkan pada	Kunci konteks kondisi global
				aws: SourceOrgID aws: SourceOrgPath aws: SourceAccount
Sumber daya	Identitas Anda hanya dapat mengakses sumber daya tepercaya	SCP	Identitas	aws: ResourceOrg ID aws: ResourceOrgPaths
	Hanya sumber daya tepercaya yang dapat diakses dari jaringan Anda	Kebijakan titik akhir VPC	Jaringan	aws: ResourceAccount

Perimeter	Tujuan kontrol	Penggunaan	Diterapkan pada	Kunci konteks kondisi global
Jaringan	Identitas Anda hanya dapat mengakses sumber daya dari jaringan yang diharapkan	SCP	Identitas	aws: SourceIp aws: SourceVpc aws: SourceVpce aws: VpceAccount aws: VpceOrgPaths aws: VpceOrg ID aws:ViaAWSService aws: PrincipalIsAwsService
	Sumber daya Anda hanya dapat diakses dari jaringan yang diharapkan	RCP	Sumber daya	

Anda dapat menganggap perimeter data sebagai menciptakan batas yang kuat di sekitar data Anda untuk mencegah pola akses yang tidak diinginkan. Meskipun perimeter data dapat mencegah akses luas yang tidak diinginkan, Anda masih perlu membuat keputusan kontrol akses yang halus. Menetapkan perimeter data tidak mengurangi kebutuhan untuk terus menyempurnakan izin dengan menggunakan alat seperti IAM Access Analyzer sebagai bagian dari perjalanan Anda ke hak istimewa yang paling sedikit.

Untuk menerapkan kontrol perimeter data pada sumber daya yang saat ini tidak didukung oleh RCP, Anda dapat menggunakan kebijakan berbasis sumber daya yang dilampirkan ke sumber daya secara langsung. Untuk daftar layanan yang mendukung RCP dan kebijakan berbasis sumber daya, lihat Kebijakan kontrol sumber daya (RCP) dan. AWS layanan yang bekerja dengan IAM

Untuk menerapkan kontrol perimeter jaringan, sebaiknya gunakan`aws:VpceOrgID`,`aws:VpceOrgPaths`, dan `aws:VpceAccount` hanya jika semua layanan yang ingin Anda batasi aksesnya saat ini didukung. Menggunakan kunci kondisi ini dengan layanan

yang tidak didukung dapat menyebabkan hasil otorisasi yang tidak diinginkan. Untuk daftar layanan yang mendukung kunci, lihatAWS kunci konteks kondisi global. Jika Anda perlu menerapkan kontrol pada berbagai layanan yang lebih luas, pertimbangkan untuk menggunakan `aws:SourceVpc` dan sebagai `aws:SourceVpce` gantinya.

Perimeter identitas

Perimeter identitas adalah seperangkat kontrol akses pencegahan kasar yang membantu memastikan hanya identitas tepercaya yang dapat mengakses sumber daya Anda dan hanya identitas tepercaya yang diizinkan dari jaringan Anda. Identitas tepercaya biasanya mencakup prinsipal (peran atau pengguna) di AWS akun dan AWS layanan Anda yang bertindak atas nama Anda. Semua identitas lainnya dianggap tidak dipercaya dan dicegah oleh perimeter identitas kecuali pengecualian eksplisit diberikan.

Kunci kondisi global berikut membantu menegakkan kontrol perimeter identitas berdasarkan definisi identitas tepercaya Anda. Gunakan kunci ini dalam kebijakan kontrol sumber daya untuk membatasi akses ke sumber daya, atau dalam kebijakan titik akhir VPC untuk membatasi akses ke jaringan Anda.

Identitas yang dimiliki oleh Anda

Anda dapat menggunakan kunci kondisi berikut untuk menentukan prinsip-prinsip IAM yang Anda buat dan kelola di. Akun AWS

- aws: PrincipalOrg ID- Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa prinsipal IAM yang membuat permintaan milik organisasi yang ditentukan di. AWS Organizations
- aws: PrincipalOrgPaths- Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa pengguna IAM, peran IAM, prinsipal pengguna AWS STS federasi, prinsipal federasi SAML, prinsipal federasi OIDC, atau Pengguna root akun AWS membuat permintaan milik unit organisasi tertentu (OU) di. AWS Organizations
- aws: PrincipalAccount— Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya hanya dapat diakses oleh akun utama yang Anda tentukan dalam kebijakan.

Identitas dari AWS layanan yang bertindak atas nama Anda

Anda dapat menggunakan kunci kondisi berikut untuk mengizinkan AWS layanan menggunakan identitas mereka sendiri untuk mengakses sumber daya Anda saat mereka bertindak atas nama Anda.

- aws: PrincipalIsAWSServicedan aws: SourceOrg ID (atau aw: SourceOrgPaths danaw: SourceAccount) — Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa ketika Layanan AWS kepala sekolah mengakses sumber daya Anda, mereka melakukannya hanya atas nama sumber daya di organisasi tertentu, unit organisasi, atau akun di AWS Organizations.

Untuk informasi selengkapnya, lihat Men etapkan perimeter data pada AWS: Izinkan hanya identitas tepercaya untuk mengakses data perusahaan.

Perimeter sumber daya

Perimeter sumber daya adalah sekumpulan kontrol akses pencegahan kasar yang membantu memastikan identitas Anda hanya dapat mengakses sumber daya tepercaya dan hanya sumber daya tepercaya yang dapat diakses dari jaringan Anda. Sumber daya tepercaya biasanya mencakup sumber daya yang dimiliki oleh AWS akun Anda atau oleh AWS layanan yang bertindak atas nama Anda.

Kunci kondisi global berikut membantu menerapkan kontrol perimeter sumber daya berdasarkan definisi sumber daya tepercaya Anda. Gunakan kunci ini di Kebijakan kontrol layanan (SCP) untuk membatasi sumber daya mana yang dapat diakses oleh identitas Anda, atau dalam kebijakan titik akhir VPC untuk membatasi sumber daya mana yang dapat diakses dari jaringan Anda.

Sumber daya yang dimiliki oleh Anda

Anda dapat menggunakan kunci kondisi berikut untuk menentukan AWS sumber daya yang Anda buat dan kelola di Akun AWS.

- aws: ResourceOrg ID- Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang diakses milik organisasi yang ditentukan di AWS Organizations.
- aw: ResourceOrgPaths- Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang diakses milik unit organisasi (OU) yang ditentukan di AWS Organizations.
- aw: ResourceAccount- Anda dapat menggunakan kunci kondisi ini untuk memastikan sumber daya yang diakses milik yang ditentukan Akun AWS.

Sumber daya dari AWS layanan yang bertindak atas nama Anda

Dalam beberapa kasus, Anda mungkin perlu mengizinkan akses ke sumber daya yang AWS dimiliki, sumber daya yang bukan milik organisasi Anda dan yang diakses oleh kepala sekolah Anda

atau oleh AWS layanan yang bertindak atas nama Anda. Untuk informasi selengkapnya tentang skenario ini, lihat Menetapkan perimeter data pada AWS: Izinkan hanya sumber daya tepercaya dari organisasi saya.

Perimeter jaringan

Perimeter jaringan adalah seperangkat kontrol akses preventif kasar yang membantu memastikan identitas Anda dapat mengakses sumber daya hanya dari jaringan yang diharapkan dan sumber daya Anda hanya dapat diakses dari jaringan yang diharapkan. Jaringan yang diharapkan biasanya mencakup pusat data lokal dan cloud pribadi virtual (VPC) dan jaringan AWS layanan yang bertindak atas nama Anda.

Kunci kondisi global berikut membantu menegakkan kontrol perimeter jaringan berdasarkan definisi jaringan yang diharapkan. Gunakan kunci ini dalam kebijakan kontrol layanan (SCP) untuk membatasi jaringan yang dapat dikomunikasikan identitas Anda, atau dalam kebijakan kontrol sumber daya (RCP) untuk membatasi akses sumber daya ke jaringan yang diharapkan.

Jaringan yang dimiliki oleh Anda

Anda dapat menggunakan kunci kondisi berikut untuk menentukan jaringan yang diharapkan digunakan karyawan dan aplikasi Anda untuk mengakses sumber daya Anda, seperti jangkauan CIDR IP perusahaan dan VPC Anda.

- aws: SourceIp— Anda dapat menggunakan kunci kondisi ini untuk memastikan alamat IP pemohon berada dalam rentang IP yang ditentukan.
- aws: SourceVpc- Anda dapat menggunakan kunci kondisi ini untuk memastikan titik akhir VPC yang dilalui permintaan milik VPC yang ditentukan.
- aws: SourceVpce- Anda dapat menggunakan kunci kondisi ini untuk memastikan permintaan berjalan melalui titik akhir VPC yang ditentukan.
- aws: VpceAccount— Anda dapat menggunakan kunci kondisi ini untuk memastikan permintaan datang melalui titik akhir VPC yang dimiliki oleh akun yang ditentukan AWS .
- aws: VpceOrgPaths— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa permintaan datang melalui titik akhir VPC yang dimiliki oleh akun milik unit organisasi tertentu (OU) di. AWS Organizations
- aws: VpceOrg ID- Anda dapat menggunakan kunci kondisi ini untuk memastikan permintaan datang melalui titik akhir VPC yang dimiliki oleh akun di organisasi yang ditentukan di. AWS Organizations

`aws:VpceAccount`,`aws:VpceOrgPaths`, dan `aws:VpceOrgID` sangat berguna untuk mengimplementasikan kontrol perimeter jaringan yang berskala secara otomatis dengan penggunaan titik akhir VPC Anda, tanpa memerlukan pembaruan kebijakan saat Anda membuat titik akhir baru. Lihat AWS kunci konteks kondisi global untuk daftar Layanan AWS yang mendukung kunci ini.

Jaringan dari AWS layanan yang bertindak atas nama Anda

Anda dapat menggunakan kunci kondisi berikut untuk mengizinkan AWS layanan mengakses sumber daya Anda dari jaringan mereka saat mereka bertindak atas nama Anda.

- aws:ViaAWSService— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa Layanan AWS dapat membuat permintaan atas nama kepala sekolah Anda menggunakan Teruskan sesi akses (FAS).
- aws: PrincipalIsAWSService— Anda dapat menggunakan kunci kondisi ini untuk memastikan bahwa Layanan AWS dapat mengakses sumber daya Anda menggunakanAWS prinsipal layanan.

Ada skenario tambahan di mana Anda perlu mengizinkan akses untuk Layanan AWS mengakses sumber daya Anda dari luar jaringan Anda. Untuk informasi selengkapnya, lihat Menetapkan perimeter data pada AWS: Mengizinkan akses ke data perusahaan hanya dari jaringan yang diharapkan.

Sumber daya untuk mempelajari lebih lanjut tentang perimeter data

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang batas data di seluruh AWS.

- Perimeter data aktif AWS — Pelajari tentang perimeter data dan manfaat serta kasus penggunaannya.
- Seri Posting Blog: Menetapkan Perimeter Data AWS - Posting blog ini mencakup panduan preskriptif tentang menetapkan perimeter data Anda dalam skala besar, termasuk pertimbangan keamanan dan implementasi utama.
- Contoh kebijakan perimeter data — GitHub Repositori ini berisi contoh kebijakan yang mencakup beberapa pola umum untuk membantu Anda menerapkan perimeter data AWS.
- Pembantu perimeter data — Alat ini membantu Anda merancang dan mengantisipasi dampak kontrol perimeter data Anda dengan menganalisis aktivitas akses di AWS CloudTrail log Anda.

Batas izin untuk entitas IAM

AWS mendukung batas izin untuk entitas IAM (pengguna atau peran). Batas izin adalah fitur lanjutan untuk menggunakan kebijakan terkelola untuk menyetel izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM. Batas izin entitas mengizinkannya untuk melakukan hanya tindakan yang diizinkan oleh kebijakan berbasis identitas dan batas izinnya.

Untuk informasi lebih lanjut tentang jenis kebijakan, lihat Jenis kebijakan.

> **Important**
>
> Jangan gunakan pernyataan kebijakan berbasis sumber daya yang menyertakan elemen `NotPrincipal` kebijakan dengan `Deny` efek untuk pengguna IAM atau peran yang memiliki kebijakan batas izin yang dilampirkan. `NotPrincipal`Elemen dengan `Deny` efek akan selalu menolak prinsipal IAM yang memiliki kebijakan batas izin terlampir, terlepas dari nilai yang ditentukan dalam `NotPrincipal` elemen. Hal ini menyebabkan beberapa pengguna atau peran IAM yang seharusnya memiliki akses ke sumber daya kehilangan akses.
> Sebaiknya ubah pernyataan kebijakan berbasis sumber daya untuk menggunakan operator kondisi `ArnNotEquals` dengan kunci `aws:PrincipalArn` konteks untuk membatasi akses alih-alih elemen. `NotPrincipal` Untuk informasi tentang `NotPrincipal` elemen, lihatAWS Elemen kebijakan JSON: NotPrincipal.

Anda dapat menggunakan kebijakan AWS terkelola atau kebijakan yang dikelola pelanggan untuk menetapkan batas untuk entitas IAM (pengguna atau peran). Kebijakan tersebut membatasi izin maksimum bagi pengguna atau peran.

Misalnya, asumsikan bahwa pengguna IAM bernama `Shirley` harus diizinkan untuk mengelola hanya Amazon S3, Amazon CloudWatch, dan Amazon EC2. Untuk menegakkan aturan ini, Anda dapat menggunakan kebijakan berikut untuk mengatur batas izin untuk pengguna `Shirley`:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Saat Anda menggunakan kebijakan untuk mengatur batas izin bagi pengguna, itu membatasi izin pengguna tetapi tidak memberikan izin sendiri. Dalam contoh ini, kebijakan menetapkan izin maksimum untuk semua `Shirley` operasi di Amazon S3, CloudWatch, dan Amazon EC2. Shirley tidak pernah dapat menjalankan operasi di layanan lain, termasuk IAM, bahkan jika dia memiliki kebijakan izin yang mengizinkannya. Misalnya, Anda dapat menambahkan kebijakan berikut ke pengguna `Shirley`:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Kebijakan ini memungkinkan pembuatan pengguna di IAM. Jika Anda melampirkan kebijakan izin ini ke pengguna `Shirley` dan Shirley mencoba membuat sebuah pengguna, operasi gagal. Itu gagal karena batas izin tidak mengizinkan operasi `iam:CreateUser`. Mengingat kedua kebijakan ini, Shirley tidak memiliki izin untuk melakukan operasi apa pun di AWS. Anda harus menambahkan kebijakan izin yang berbeda untuk mengizinkan tindakan di layanan lain, seperti Amazon S3. Atau, Anda dapat memperbarui batas izin untuk memungkinkannya membuat pengguna di IAM.

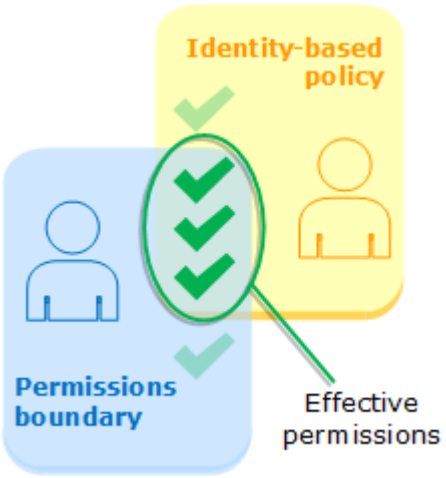
Mengevaluasi izin efektif dengan batasan

Batas izin untuk entitas IAM (pengguna atau peran) menetapkan izin maksimum yang dapat dimiliki oleh entitas. Ini dapat mengubah izin efektif bagi pengguna atau peran tersebut. Izin efektif untuk

sebuah entitas adalah izin yang diberikan oleh semua kebijakan yang memengaruhi pengguna atau peran. Dalam akun, izin untuk sebuah entitas dapat dipengaruhi oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, batasan izin, SCP AWS Organizations , atau kebijakan sesi. Untuk informasi selengkapnya tentang berbagai jenis kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

Jika salah satu dari jenis kebijakan ini secara jelas menolak akses untuk suatu operasi, maka permintaan tersebut ditolak. Izin yang diberikan ke suatu entitas oleh beberapa jenis izin jauh lebih rumit. Untuk detail selengkapnya tentang cara AWS mengevaluasi kebijakan, lihatLogika evaluasi kebijakan.

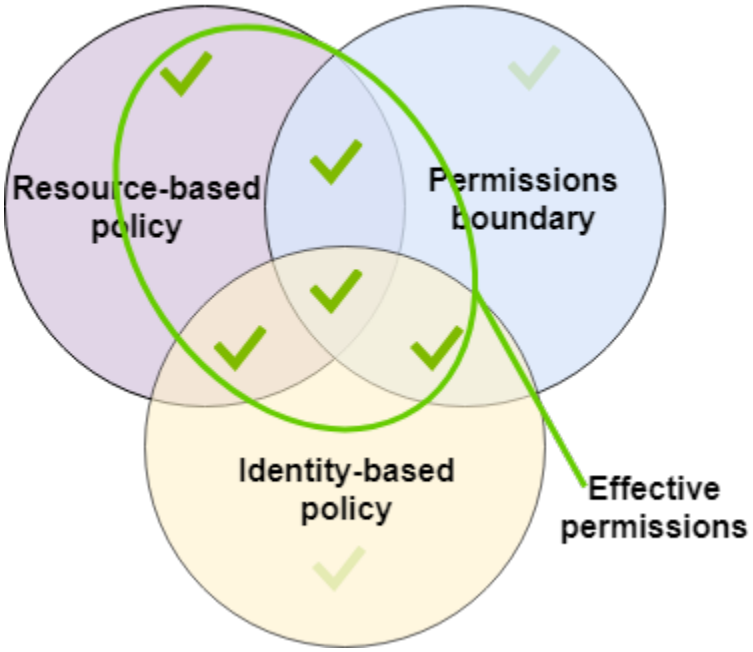
Identity-based kebijakan dengan batas — Identity-based kebijakan adalah kebijakan sebaris atau terkelola yang dilampirkan ke pengguna, grup pengguna, atau peran. Identity-based kebijakan memberikan izin kepada entitas, dan batas izin membatasi izin tersebut. Izin yang efektif adalah titik temu dari kedua jenis kebijakan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



Resource-based kebijakan — Resource-based kebijakan mengontrol bagaimana prinsipal yang ditentukan dapat mengakses sumber daya yang dilampirkan kebijakan.

Resource-based kebijakan untuk pengguna IAM

> Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna IAM ARN (yang bukan sesi utama pengguna AWS STS federasi) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.



Resource-based kebijakan untuk peran IAM

Peran IAM — Resource-based kebijakan yang memberikan izin ke peran IAM ARN dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

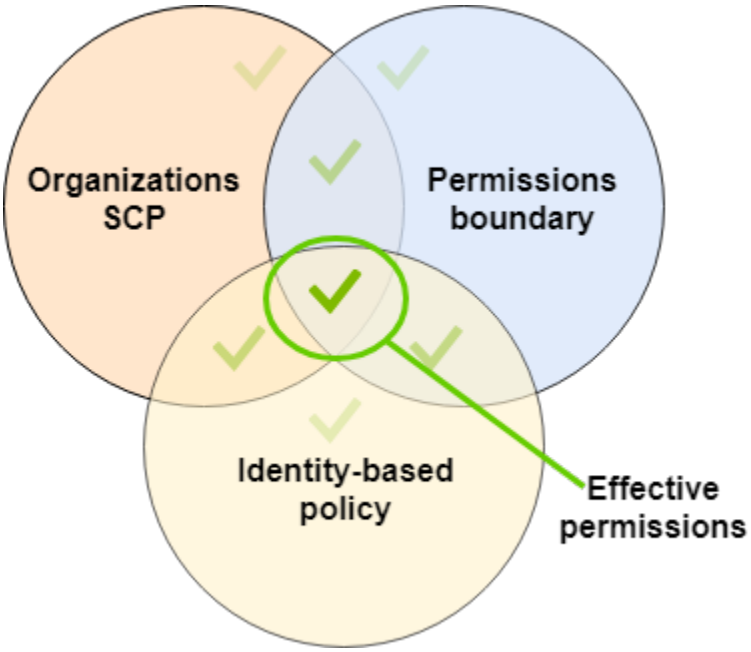
Sesi peran IAM — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin ke sesi peran IAM ARN memberikan izin langsung ke sesi peran yang diasumsikan. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. Ketika Anda mengambil peran dan membuat permintaan, prinsipal yang membuat permintaan adalah sesi peran IAM ARN dan bukan ARN dari peran itu sendiri.

Resource-based kebijakan untuk AWS STS sesi utama pengguna federasi

AWS STS sesi utama pengguna federasi — Sesi utama AWS STS pengguna federasi adalah sesi yang dibuat dengan panggilan. `GetFederationToken` Ketika pengguna federasi membuat permintaan, prinsipal yang membuat permintaan adalah pengguna federasi ARN dan bukan ARN dari pengguna IAM yang berfederasi. Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna federasi ARN memberikan izin langsung ke sesi. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi.

Namun, jika kebijakan berbasis sumber daya memberikan izin kepada ARN pengguna IAM yang melakukan federasi, maka permintaan yang dibuat oleh prinsipal pengguna AWS STS federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

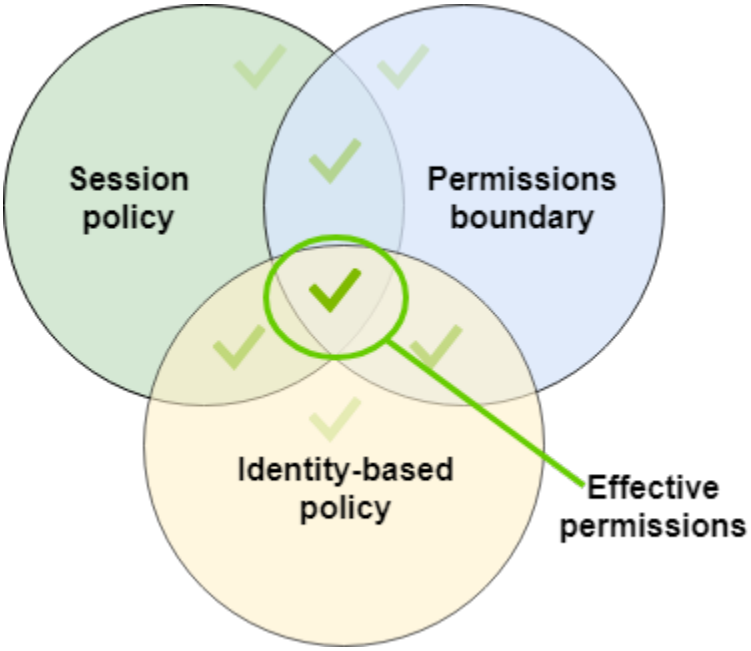
AWS Organizations SCP — SCP diterapkan ke keseluruhan. Akun AWS Mereka membatasi izin untuk setiap permintaan yang dibuat oleh prinsipal di dalam akun. Sebuah entitas IAM (pengguna atau peran) dapat membuat permintaan yang terdampak oleh SCP, batas izin, dan kebijakan berbasis identitas. Dalam hal ini, permintaan hanya diizinkan jika ketiga jenis kebijakan mengizinkannya. Izin yang efektif adalah titik temu dari ketiga jenis kebijakan. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin.



Anda dapat mempelajari apakah akun Anda adalah anggota organisasi di AWS Organizations. Anggota organisasi dapat dipengaruhi oleh SCP. Untuk melihat data ini menggunakan AWS CLI perintah atau operasi AWS API, Anda harus memiliki izin untuk `organizations:DescribeOrganization` tindakan untuk AWS Organizations entitas Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di AWS Organizations konsol. Untuk mengetahui apakah SCP menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi AWS Organizations administrator Anda.

Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter saat Anda secara terprogram membuat sesi sementara untuk peran atau pengguna gabungan. Izin untuk sesi berasal dari entitas IAM (pengguna atau peran) yang digunakan untuk membuat sesi dan dari kebijakan sesi. Izin kebijakan berbasis identitas milik entitas akan dibatasi oleh kebijakan

sesi dan batasan izin. Izin yang efektif untuk set jenis kebijakan ini adalah titik temu dari ketiga jenis kebijakan. Penolakan secara eksplisit dalam salah satu kebijakan ini membatalkan izin. Untuk informasi selengkapnya tentang kebijakan sesi, lihat Kebijakan Sesi



Mendelegasikan tanggung jawab kepada orang lain menggunakan batas izin

Anda dapat menggunakan batasan izin untuk memberikan tugas manajemen perizinan, seperti pembuatan pengguna, ke pengguna IAM di akun Anda. Hal ini mengizinkan orang lain melakukan tugas atas nama Anda dalam batas izin tertentu.

Misalnya, asumsikan bahwa María adalah administrator dari X-Company Akun AWS. Dia ingin memberikanan tugas pembuatan pengguna ke Zhang. Namun, ia harus memastikan bahwa Zhang membuat pengguna yang mematuhi aturan perusahaan berikut:

- Pengguna tidak dapat menggunakan IAM untuk membuat atau mengelola pengguna, grup, peran, atau kebijakan.
- Pengguna tidak diberi akses ke `logs` bucket Amazon S3 dan tidak dapat mengakses instans Amazon EC2 `i-1234567890abcdef0`.
- Pengguna tidak dapat menghapus kebijakan batas milik mereka.

Untuk menegakkan aturan ini, María menyelesaikan tugas berikut, yang perinciannya tercantum di bawah ini:

1. María membuat kebijakan terkelola `XCompanyBoundaries` untuk digunakan sebagai batas izin bagi semua pengguna baru dalam akun.
2. María membuat kebijakan terkelola `DelegatedUserBoundary` dan menetapkannya sebagai batas izin untuk Zhang. Maria mencatat ARN pengguna admin -nya dan menggunakannya dalam kebijakan untuk mencegah Zhang mengaksesnya.
3. María membuat kebijakan terkelola `DelegatedUserPermissions` dan melampirkannya sebagai kebijakan izin untuk Zhang.
4. María memberi tahu Zhang tentang tanggung jawab dan batasan barunya.

Tugas 1: María harus terlebih dahulu membuat kebijakan terkelola untuk menentukan batas bagi pengguna baru. María akan mengizinkan Zhang untuk memberikan kebijakan izin yang dibutuhkan pengguna, tetapi dia ingin pengguna tersebut dibatasi. Untuk melakukan ini, ia membuat kebijakan terkelola pelanggan berikut ini dengan nama `XCompanyBoundaries`. Kebijakan ini melakukan hal-hal berikut:

- Mengizinkan pengguna akses penuh ke beberapa layanan
- Mengizinkan penggelolaan mandiri terbatas di konsol IAM. Ini berarti mereka dapat mengubah kata sandi mereka setelah masuk ke konsol. Mereka tidak dapat mengatur kata sandi awal mereka. Untuk mengijinkan ini, tambahkan tindakan `"*LoginProfile"` ke pernyataan `AllowManageOwnPasswordAndAccessKeys`.
- Menolak akses pengguna ke catatan bucket Amazon S3 atau instans Amazon EC2 `i-1234567890abcdef0`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. Per `ServiceBoundaries` nyataan kebijakan ini memungkinkan akses penuh ke AWS layanan yang ditentukan. Artinya jika tindakan pengguna baru dalam layanan ini hanya dibatasi oleh kebijakan izin yang terlampir pada pengguna.
2. Pernyataan `AllowIAMConsoleForCredentials` ini mengizinkan akses untuk mendaftar semua pengguna IAM. Akses ini diperlukan untuk menavigasi halaman Pengguna di Konsol Manajemen AWS. Ini juga mengijinkan untuk melihat persyaratan kata sandi untuk akun, yang diperlukan saat mengubah kata sandi Anda sendiri.
3. Per `AllowManageOwnPasswordAndAccessKeys` nyataan ini memungkinkan pengguna untuk mengelola hanya kata sandi konsol mereka sendiri dan kunci akses terprogram. Ini penting jika Zhang atau administrator lain menetapkan kebijakan izin kepada pengguna baru dengan akses IAM penuh. Dalam kasus ini, pengguna tersebut kemudian dapat mengubah izin miliknya sendiri atau milik pengguna lain. Pernyataan ini mencegah hal tersebut terjadi.
4. Pernyataan `DenyS3Logs` secara jelas menolak akses ke bucket `logs`.
5. Pernyataan `DenyEC2Production` secara jelas menolak akses ke instans `i-1234567890abcdef0`.

Tugas 2: María ingin mengizinkan Zhang membuat semua X-Company pengguna, tetapi hanya dengan batas `XCompanyBoundaries` izin. Dia menciptakan kebijakan terkelola pelanggan berikut bernama `DelegatedUserBoundary`. Kebijakan ini menentukan izin maksimum yang dapat dimiliki Zhang.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
```

```
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
              "iam:ListUsers",
              "iam:SetDefaultPolicyVersion",
```

```
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy",
                "iam:UpdateGroup",
                "iam:UpdateLoginProfile",
                "iam:UpdateUser"
            ],
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. Pernyataan `CreateOrChangeOnlyWithBoundary` mengizinkan Zhang untuk membuat pengguna IAM tetapi hanya jika dia menggunakan kebijakan `XCompanyBoundaries` untuk menetapkan batas izin. Pernyataan ini juga mengizinkannya untuk menetapkan batas izin bagi pengguna yang ada tetapi hanya menggunakan kebijakan yang sama. Terakhir, pernyataan ini memungkinkan Zhang untuk mengelola kebijakan izin bagi pengguna dengan set batas izin ini.
2. Pernyataan `CloudWatchAndOtherIAMTasks` mengizinkan Zhang untuk menyelesaikan tugas pengguna, grup, dan kebijakan lainnya. Dia memiliki izin untuk mengatur ulang kata sandi dan membuat kunci akses untuk setiap pengguna IAM yang tidak terdaftar dalam elemen

`NotResource` kebijakan. Ini memungkinkannya untuk membantu pengguna dengan masalah masuk.

3. Pernyataan `NoBoundaryPolicyEdit` menolak akses Zhang untuk memperbarui kebijakan `XCompanyBoundaries`. Dia tidak diperbolehkan mengubah kebijakan apa pun yang digunakan untuk menetapkan batas izin bagi dirinya sendiri atau pengguna lain.
4. Pernyataan `NoBoundaryUserDelete` menolak akses Zhang untuk menghapus batas izin bagi dirinya sendiri atau pengguna lain.

Kemudian María menetapkan kebijakan `DelegatedUserBoundary` sebagai batas izin untuk pengguna `Zhang`.

Tugas 3: Karena batas izin membatasi izin maksimum, tetapi tidak memberikan akses sendiri, Maria harus membuat kebijakan izin untuk Zhang. Dia menciptakan kebijakan berikut bernama `DelegatedUserPermissions`. Kebijakan ini menetapkan operasi yang dapat dilakukan Zhang, dalam batas izin yang ditentukan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

Setiap pernyataan memiliki tujuan yang berbeda:

1. Pernyataan `IAM` kebijakan ini mengizinkan akses penuh Zhang ke IAM. Namun, karena batas izinnya hanya mengizinkan beberapa operasi IAM, izin efektif IAM-nya dibatasi hanya oleh batas izinnya.
2. Per `CloudWatchLimited` nyataan itu memungkinkan Zhang untuk melakukan lima tindakan di CloudWatch. Batas izinnya memungkinkan semua tindakan masuk CloudWatch, sehingga CloudWatch izin efektifnya hanya dibatasi oleh kebijakan izinnya.
3. Pernyataan `S3BucketContents` mengizinkan Zhang membuat daftar Amazon S3 bucket `ZhangBucket`. Namun, batas izinnya tidak mengizinkan tindakan Amazon S3 apapun, sehingga dia tidak dapat melakukan operasi S3 apapun, terlepas dari kebijakan perizinannya.

> **Note**
>
> Kebijakan Zhang mengizinkannya untuk membuat pengguna yang kemudian dapat mengakses sumber daya Amazon S3 yang tidak dapat diaksesnya. Dengan menyerahkan tindakan administratif ini, Maria secara efektif memercayai Zhang dengan akses ke Amazon S3

Kemudian María menetapkan kebijakan `DelegatedUserPermissions` sebagai kebijakan izin untuk pengguna `Zhang`.

Tugas 4: Dia memberikan Zhang instruksi untuk membuat pengguna baru. Dia mengatakan padanya bahwa dia dapat membuat pengguna baru dengan izin apapun yang mereka butuhkan, tetapi ia harus menetapkan untuknya kebijakan `XCompanyBoundaries` sebagai batas izin.

Zhang menyelesaikan tugas berikut:

1. Zhang membuat pengguna dengan Konsol Manajemen AWS. Dia mengetik nama pengguna `Nikhil` dan mengaktifkan akses konsol untuk pengguna. Dia menghapus kotak centang di samping Perlu pengaturan ulang kata sandi, karena kebijakan di atas mengizinkan pengguna untuk mengubah kata sandi mereka hanya setelah mereka masuk ke konsol IAM.
2. Di halaman Atur perizinan, Zhang memilih kebijakan izin IAMFullAccess dan AmazonS3ReadOnlyAccess yang mengizinkan Nikhil melakukan pekerjaannya.
3. Zhang melewatkan bagian Atur batasan izin, melupakan instruksi María.
4. Zhang meninjau detail pengguna dan memilih Buat pengguna.

   Operasi gagal dan akses ditolak. Batas izin Zhang `DelegatedUserBoundary` mengharuskan setiap pengguna yang dia buat memiliki kebijakan `XCompanyBoundaries` yang digunakan sebagai batas izin.
5. Zhang kembali ke halaman sebelumnya. Di bagian Atur batasan izin, ia memilih kebijakan `XCompanyBoundaries`.
6. Zhang meninjau detail pengguna dan memilih Buat pengguna.

   Pengguna dibuat.

Ketika Nikhil masuk, dia memiliki akses ke IAM dan Amazon S3, kecuali operasi yang ditolak oleh batas izin. Misalnya, ia dapat mengubah kata sandinya sendiri di IAM tetapi tidak dapat membuat pengguna lain atau mengubah kebijakannya. Nikhil memiliki akses hanya baca ke Amazon S3

Jika seseorang menambahkan kebijakan berbasis sumber daya ke bucket `logs` yang mengizinkan Nikhil memasukkan sebuah objek ke dalam bucket, dia tetap tidak dapat mengakses bucket. Alasannya adalah bahwa tindakan apa pun di bucket `logs` secara jelas ditolak oleh batas izinnya. Penolakan secara jelas dalam jenis kebijakan apa pun menyebabkan permintaan ditolak.

Namun, jika kebijakan berbasis sumber daya yang terlampir pada rahasia Secrets Manager memungkinkan Nikhil untuk melakukan tindakan `secretsmanager:GetSecretValue`, lalu Nikhil dapat mengambil dan mendekripsi rahasia. Alasannya adalah bahwa operasi Secrets Manager tidak secara jelas ditolak oleh batas izinnya, dan penolakan implisit dalam batas izin tidak membatasi kebijakan berbasis sumber daya.

Identity-based kebijakan dan kebijakan berbasis sumber daya

Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. Saat Anda membuat kebijakan izin untuk membatasi akses ke sumber daya, Anda dapat memilih kebijakan berbasis identitas atau kebijakan berbasis sumber daya.

Identity-based kebijakan dilampirkan ke pengguna, grup, atau peran IAM. Kebijakan ini memungkinkan Anda menentukan apa yang dapat dilakukan oleh identitas (izinnya). Misalnya, Anda dapat melampirkan kebijakan ke pengguna IAM bernama John, yang menyatakan bahwa dia diizinkan untuk melakukan tindakan `RunInstances` Amazon EC2. Kebijakan selanjutnya dapat menyatakan bahwa John diizinkan untuk mendapatkan item dari tabel Amazon DynamoDB bernama. `MyCompany` Anda juga dapat mengizinkan John untuk mengelola kredensyal keamanan IAM-nya sendiri. Identity-based kebijakan dapat dikelola atau inline.

Resource-based kebijakan dilampirkan ke sumber daya. Misalnya, Anda dapat melampirkan kebijakan berbasis sumber daya ke bucket Amazon S3, antrian Amazon SQS, titik akhir VPC, kunci enkripsi, tabel dan aliran Amazon DynamoDB, serta sumber daya. AWS Key Management Service AWS Sign-In Untuk daftar layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM.

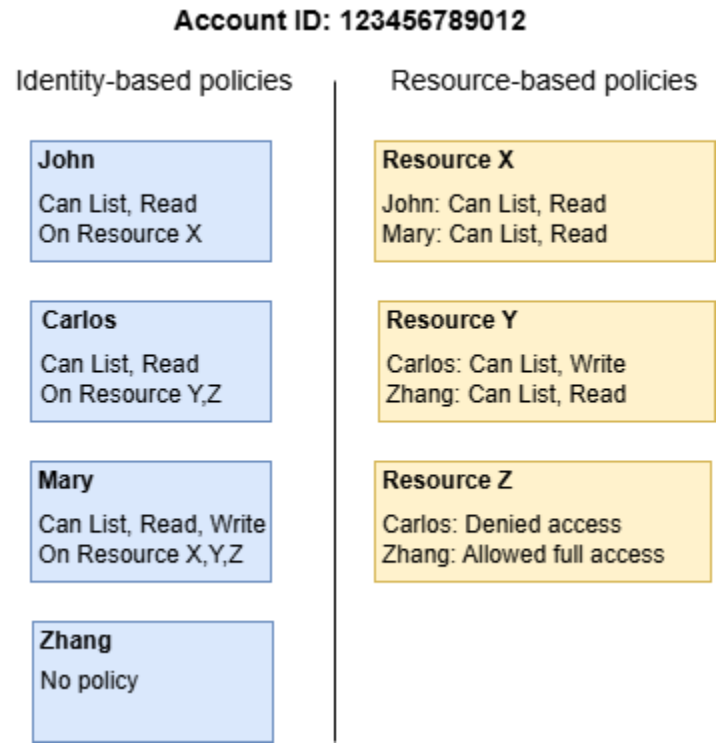
Dengan kebijakan berbasis sumber daya, Anda dapat menentukan siapa yang memiliki akses ke sumber daya tersebut dan tindakan apa yang dapat mereka lakukan di situ. Untuk mengetahui apakah prinsipal di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Apa itu IAM Access Analyzer? . Resource-based kebijakan hanya inline, tidak dikelola.

> **Note**
>
> Resource-basedkebijakan berbeda dari izin tingkat sumber daya. Anda dapat melampirkan kebijakan berbasis sumber daya secara langsung ke sumber daya, seperti yang dijelaskan dalam topik ini. Resource-level izin mengacu pada kemampuan untuk menggunakan ARN untuk menentukan sumber daya individu dalam kebijakan. Resource-based Kebijakan hanya didukung oleh beberapa AWS layanan. Untuk daftar yang layanannya didukung kebijakan berbasis sumber daya dan izin tingkat sumber daya, lihat AWS layanan yang bekerja dengan IAM.

Untuk mempelajari cara berinteraksi antara kebijakan berbasis identitas dan kebijakan berbasis sumber daya dalam akun yang sama, lihat Evaluasi kebijakan untuk permintaan dalam satu akun.

Untuk mempelajari cara interaksi kebijakan di seluruh akun, lihat Cross-account logika evaluasi kebijakan.

Untuk lebih memahami konsep ini, lihat gambar berikut. Administrator dari akun `123456789012` terlampir kebijakan berbasis identitas kepada pengguna `John`, `Carlos`, dan `Mary`. Beberapa tindakan dalam kebijakan ini dapat dilakukan pada sumber daya tertentu. Misalnya, pengguna `John` dapat melakukan beberapa tindakan di `Resource X`. Ini adalah izin tingkat sumber daya dalam kebijakan berbasis identitas. Administrator juga menambahkan kebijakan berbasis sumber daya ke`Resource X`,, dan. `Resource Y` `Resource Z` Resource-based kebijakan memungkinkan Anda menentukan siapa yang dapat mengakses sumber daya tersebut. Misalnya, kebijakan berbasis sumber daya di `Resource X` mengizinkan akses pengguna `John` dan `Mary` daftar dan baca ke sumber daya.



Contoh akun `123456789012` mengizinkan pengguna berikut untuk melakukan tindakan yang tercantum:

- John — John dapat melakukan daftar dan membaca tindakan`Resource X`. Dia diberikan izin ini oleh kebijakan berbasis identitas pada penggunanya dan kebijakan berbasis sumber daya pada `Resource X`.
- Carlos — Carlos dapat melakukan daftar, membaca, dan menulis tindakan`Resource Y`, tetapi ditolak aksesnya. `Resource Z` Kebijakan berbasis identitas pada Carlos memungkinkan dia

untuk melakukan tindakan daftar dan baca `Resource Y`. Kebijakan berbasis sumber daya `Resource Y` juga memungkinkan dia menulis izin. Namun, meskipun kebijakan berbasis identitas miliknya mengizinkannya mengakses `Resource Z`, kebijakan berbasis sumber daya `Resource Z` menolak akses tersebut. `Deny` secara jelas menimpa `Allow` dan aksesnya ke `Resource Z` ditolak. Untuk informasi selengkapnya, lihat Logika evaluasi kebijakan.

- Mary — Mary dapat melakukan operasi daftar, membaca, dan menulis pada`Resource X`,`Resource Y`, dan`Resource Z`. Kebijakan berbasis identitas miliknya mengizinkan tindakan yang lebih banyak terhadap lebih banyak sumber daya daripada kebijakan berbasis sumber daya, tetapi tidak ada yang menolak akses.
- Zhang — Zhang memiliki akses penuh ke. `Resource Z` Zhang tidak memiliki kebijakan berbasis identitas, tetapi kebijakan berbasis sumber daya `Resource Z` mengizinkannya mengakses sumber daya sepenuhnya. Zhang juga dapat melakukan tindakan daftar dan baca pada `Resource Y`.

Identity-based kebijakan dan kebijakan berbasis sumber daya adalah kebijakan izin dan dievaluasi bersama. Untuk permintaan yang hanya berlaku kebijakan izin, AWS pertama-tama periksa semua kebijakan untuk file. `Deny` Jika ada, maka permintaan ditolak. Kemudian AWS memeriksa setiap `Allow`. Jika setidaknya satu pernyataan mengizinkan tindakan dalam permintaan tersebut, permintaan tersebut diizinkan. Tidak peduli apakah `Allow` berada dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya.

> **Important**
>
> Logika ini hanya berlaku ketika permintaan dibuat dalam satu Akun AWS. Untuk permintaan yang dibuat dari satu akun ke akun lain, pemohon di `Account A` harus memiliki kebijakan berbasis identitas yang mengizinkan mereka mengajukan permintaan kepada sumber daya di `Account B`. Juga, kebijakan berbasis sumber daya di `Account B` harus mengizinkan pemohon di `Account A` untuk mengakses sumber daya. Harus ada kebijakan di kedua akun yang mengizinkan operasi, jika tidak permintaan tersebut gagal. Untuk informasi selengkapnya tentang menggunakan kebijakan berbasis sumber daya bagi akses akun silang, lihat Akses sumber daya lintas akun di IAM.

Pengguna yang memiliki izin spesifik mungkin meminta sumber daya yang juga memiliki kebijakan izin yang terlampir. Dalam hal ini, AWS mengevaluasi kedua set izin saat menentukan apakah akan

memberikan akses ke sumber daya. Untuk informasi tentang bagaimana kebijakan dievaluasi, lihat Logika evaluasi kebijakan.

> **Note**
>
> Amazon S3 mendukung kebijakan berbasis identitas dan kebijakan berbasis sumber daya (disebut sebagai kebijakan bucket). Sebagai tambahan, Amazon S3 mendukung mekanisme izin yang dikenal sebagai Access Control List (ACL) yang terpisah dari kebijakan dan izin IAM. Anda dapat menggunakan kebijakan IAM dalam kombinasi dengan Amazon S3 ACL. Untuk informasi selengkapnya, lihat Kontrol Akses di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Kontrol akses ke AWS sumber daya menggunakan kebijakan

Anda dapat menggunakan kebijakan untuk mengontrol akses ke sumber daya dalam IAM atau semua sumber daya. AWS

Untuk menggunakan kebijakan untuk mengontrol akses AWS, Anda harus memahami cara AWS memberikan akses. AWS terdiri dari koleksi sumber daya. Pengguna IAM adalah sebuah sumber daya. Bucket Amazon S3 adalah sebuah sumber daya. Saat Anda menggunakan AWS API, file AWS CLI, atau Konsol Manajemen AWS untuk melakukan operasi (seperti membuat pengguna), Anda mengirim permintaan untuk operasi itu. Permintaan Anda menentukan tindakan, sumber daya, sebuah entitas prinsipal (pengguna atau peran), sebuah akun prinsipal, dan informasi permintaan yang diperlukan. Semua informasi ini memberikan konteks.

AWS kemudian memeriksa bahwa Anda (kepala sekolah) diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk melakukan tindakan yang ditentukan pada sumber daya yang ditentukan. Selama otorisasi, AWS periksa semua kebijakan yang berlaku untuk konteks permintaan Anda. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON dan menentukan izin untuk entitas utama. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

AWS mengotorisasi permintaan hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan. Untuk melihat diagram proses ini, lihat Cara kerja IAM. Untuk detail tentang cara AWS menentukan apakah permintaan diizinkan, lihatLogika evaluasi kebijakan.

Saat Anda membuat kebijakan IAM, Anda dapat mengontrol akses ke hal berikut:

- Prinsipal – Kontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal).
- Identitas IAM — Kontrol identitas IAM mana (grup IAM, pengguna, dan peran) yang dapat diakses dan bagaimana caranya.
- Kebijakan IAM – Kontrol siapa yang dapat membuat, mengubah, dan menghapus kebijakan yang dikelola pelanggan, dan siapa yang dapat melampirkan dan memisahkan semua kebijakan yang dikelola.
- AWS Sumber Daya — Kontrol siapa yang memiliki akses ke sumber daya menggunakan kebijakan berbasis identitas atau kebijakan berbasis sumber daya.
- AWS Akun — Kontrol apakah permintaan hanya diperbolehkan untuk anggota akun tertentu.

Kebijakan memungkinkan Anda menentukan siapa yang memiliki akses ke AWS sumber daya, dan tindakan apa yang dapat mereka lakukan pada sumber daya tersebut. Setiap pengguna IAM dimulai dengan tanpa izin. Dengan kata lain, secara default, pengguna tidak dapat melakukan apa pun, bahkan tidak dapat melihat kunci akses milik mereka. Untuk memberikan izin kepada pengguna untuk melakukan sesuatu, Anda dapat menambahkan izin kepada pengguna (yaitu, melampirkan kebijakan ke pengguna). Atau, Anda dapat menambahkan pengguna ke grup pengguna yang memiliki izin yang dimaksud.

Misalnya, Anda dapat memberikan izin pengguna untuk mencantumkan access key-nya sendiri. Anda juga dapat memperluas izin tersebut dan mengizinkan setiap pengguna membuat, memperbarui, serta menghapus kunci milik mereka.

Saat Anda memberikan izin kepada grup pengguna, semua pengguna dalam grup pengguna tersebut mendapatkan izin tersebut. Misalnya, Anda dapat memberikan izin kepada grup pengguna Administrator untuk melakukan tindakan IAM pada salah satu sumber daya. Akun AWS Contoh lain: Anda dapat memberikan izin grup pengguna Manajer untuk menjelaskan instans Amazon EC2 dari. Akun AWS

Untuk informasi tentang cara mendelegasikan izin dasar kepada pengguna, grup IAM, dan peran Anda, lihat. Izin diperlukan untuk mengakses sumber daya IAM Untuk contoh tambahan dari kebijakan yang menggambarkan izin dasar, lihat Contoh kebijakan untuk mengelola sumber daya IAM.

Mengontrol akses untuk prinsipal

Anda dapat menggunakan kebijakan untuk mengontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal). Untuk melakukannya, Anda harus melampirkan kebijakan berbasis identitas pada identitas prang tersebut (pengguna, grup pengguna, atau peran). Anda juga

dapat menggunakan batas izin untuk mengatur izin maksimum yang dapat dimiliki sebuah entitas (pengguna atau peran).

Misalnya, asumsikan bahwa Anda ingin pengguna Zhang Wei memiliki akses penuh ke, Amazon DynamoDB CloudWatch, Amazon EC2, dan Amazon S3. Anda dapat membuat dua kebijakan berbeda agar kemudian Anda dapat memisahkannya jika Anda memerlukan satu set izin untuk pengguna yang berbeda. Atau Anda dapat menggabungkan kedua izin tersebut dalam satu kebijakan, kemudian melampirkan kebijakan tersebut ke pengguna IAM yang bernama Zhang Wei. Anda juga dapat melampirkan kebijakan ke grup pengguna di mana Zhang berada, atau peran yang dapat diambil oleh Zhang. Hasilnya, saat Zhang melihat konten di bucket S3, permintaannya diizinkan. Jika dia mencoba membuat pengguna IAM baru, permintaannya ditolak karena tidak memiliki izin.

Anda dapat menggunakan batas izin pada Zhang untuk memastikan bahwa dia tidak pernah diberi akses ke bucket S3 `amzn-s3-demo-bucket1`. Untuk melakukannya, tentukan maksimum izin yang Anda inginkan Zhang untuk memilikinya. Dalam kasus ini, Anda mengontrol apa yang dia lakukan dengan menggunakan kebijakan izinnya. Di sini, Anda hanya peduli bahwa dia tidak mengakses bucket rahasia. Jadi, Anda menggunakan kebijakan berikut untuk menentukan batas Zhang untuk mengizinkan semua AWS tindakan untuk Amazon S3 dan beberapa layanan lainnya tetapi menolak akses ke bucket S3. `amzn-s3-demo-bucket1` Karena batas izin tidak mengizinkan tindakan IAM apa pun, hal itu mencegah Zhang menghapus batas miliknya (atau siapa pun).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
```

```
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1",
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

Saat Anda menetapkan kebijakan seperti ini sebagai batas izin bagi pengguna, ingatlah bahwa itu tidak memberikan izin apa pun. Itu menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM. Untuk informasi lebih lanjut tentang batas izin, lihat Batas izin untuk entitas IAM.

Untuk informasi terperinci tentang prosedur yang disebutkan sebelumnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat kebijakan IAM yang dapat Anda lampirkan ke prinsipal, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara melampirkan kebijakan IAM pada prinsipal, lihat Menambahkan dan menghapus izin identitas IAM.
- Untuk melihat contoh kebijakan untuk memberikan akses penuh ke EC2, lihat Amazon EC2: Memungkinkan akses EC2 penuh dalam Wilayah tertentu, secara terprogram dan di konsol.
- Untuk memungkinkan akses hanya-baca ke bucket S3, gunakan dua pernyataan pertama dari contoh kebijakan berikut: Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol.
- Untuk melihat contoh kebijakan yang memungkinkan pengguna menyetel kredensialnya, seperti kata sandi konsol, kunci akses terprogram, dan perangkat MFA mereka, lihat. AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan

Mengontrol akses ke identitas

Anda dapat menggunakan kebijakan IAM untuk mengontrol apa yang dapat dilakukan pengguna terhadap identitas dengan membuat kebijakan yang Anda lampirkan ke semua pengguna melalui grup pengguna. Untuk melakukan ini, buat kebijakan yang membatasi apa yang dapat dilakukan pada sebuah identitas, atau siapa yang dapat mengaksesnya.

Misalnya, Anda dapat membuat grup pengguna bernama AllUsers, lalu melampirkan grup pengguna tersebut ke semua pengguna. Saat membuat grup pengguna, Anda dapat memberikan akses kepada semua pengguna untuk mengatur kredensialnya seperti yang dijelaskan di bagian sebelumnya. Anda kemudian dapat membuat kebijakan yang menolak akses untuk mengubah grup pengguna kecuali jika nama pengguna disertakan dalam ketentuan dari kebijakan tersebut. Namun bagian dari kebijakan tersebut hanya akan menolak akses ke siapa pun kecuali pengguna yang tercantum. Anda juga harus menyertakan izin yang mengizinkan semua tindakan manajemen grup pengguna untuk semua orang di dalam grup pengguna. Terakhir, Anda melampirkan kebijakan ini ke grup pengguna sehingga itu diterapkan ke semua pengguna. Hasilnya, saat pengguna yang tidak disebutkan dalam kebijakan mencoba melakukan perubahan pada grup pengguna, permintaannya ditolak.

Untuk membuat kebijakan ini dengan editor visual

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Kebijakan.

   Jika ini pertama kalinya Anda memilih Kebijakan, akan muncul halaman Selamat Datang di Kebijakan Terkelola. Pilih Memulai.
3. Pilih Buat kebijakan.
4. Pada bagian Editor kebijakan, pilih opsi Visual.
5. Di Pilih layanan pilih IAM.
6. Di Tindakan yang diizinkan, **`group`** ketik kotak pencarian. Editor visual menunjukkan semua tindakan IAM yang mengandung kata`group`. Pilih semua kotak centang.
7. Pilih Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda. Berdasarkan tindakan yang Anda pilih, Anda akan melihat jenis sumber daya grup dan pengguna.

   - grup - Pilih Tambah ARNs. Untuk Sumber Daya di, pilih opsi Akun apa pun. Pilih kotak centang Nama grup apa pun dengan jalur lalu ketik nama **`AllUsers`** grup pengguna. Kemudian pilih Tambahkan ARNs.
   - pengguna — Pilih kotak centang di sebelah Apa saja di akun ini.

   Salah satu tindakan yang Anda pilih, `ListGroups`, tidak mendukung penggunaan sumber daya tertentu. Anda tidak harus memilih Semua sumber daya untuk tindakan itu. Saat menyimpan kebijakan atau melihat kebijakan di editor JSON, Anda dapat melihat bahwa IAM secara otomatis membuat blok izin baru yang memberikan izin tindakan ini pada semua sumber daya.

8. Untuk menambahkan blok izin lain, pilih Tambahkan izin lainnya.
9. Pilih Pilih layanan dan kemudian pilih IAM.
10. Pilih Tindakan yang diizinkan dan kemudian pilih Beralih untuk menolak izin. Saat Anda melakukannya, seluruh blok digunakan untuk menolak izin.
11. Jenis **`group`** di kotak pencarian. Editor visual menunjukkan kepada Anda semua tindakan IAM yang berisi kata`group`. Pilih kotak centang di sebelah tindakan berikut:

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup
12. Pilih Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda. Berdasarkan tindakan yang Anda pilih, Anda akan melihat jenis sumber daya grup. Pilih Tambahkan ARNs. Untuk Sumber Daya di, pilih opsi Akun apa pun. Untuk Setiap nama grup dengan jalur, ketik nama grup pengguna**`AllUsers`**. Kemudian pilih Tambahkan ARNs.
13. Pilih kondisi Permintaan - opsional dan kemudian pilih Tambahkan kondisi lain. Lengkapi formulir dengan nilai berikut:

    - Kunci kondisi - Pilih aws:username
    - Pengukur – Pilih Default
    - Operator – Pilih StringNotEquals
    - Nilai — Ketik **`srodriguez`** dan kemudian pilih Tambah untuk menambahkan nilai lain. Ketik **`mjackson`** dan kemudian pilih Tambah untuk menambahkan nilai lain. Ketik **`adesai`** dan kemudian pilih Tambahkan kondisi.

    Ketentuan ini memastikan bahwa akses akan ditolak untuk tindakan manajemen grup pengguna tertentu saat pengguna membuat panggilan tidak dimasukkan dalam daftar. Karena ini secara jelas menolak izin, ini menggantikan blok sebelumnya yang mengizinkan pengguna tersebut untuk memanggil tindakan. Pengguna dalam daftar tidak ditolak aksesnya, dan mereka diberikan

izin pada blok izin pertama, sehingga mereka dapat sepenuhnya mengelola grup pengguna tersebut.

14. Setelah selesai, pilih Selanjutnya.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di opsi Editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

15. Pada halaman Tinjau dan buat, untuk Nama Kebijakan, ketik**`LimitAllUserGroupManagement`**. Untuk bagian Description (Deskripsi), ketik **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Tinjau Izin yang ditentukan dalam kebijakan ini untuk memastikan bahwa Anda telah memberikan izin yang dimaksud. Kemudian pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.
16. Lampirkan kebijakan tersebut ke grup pengguna Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Jika tidak, Anda dapat membuat kebijakan yang sama menggunakan contoh dokumen kebijakan JSON ini. Untuk melihat kebijakan JSON ini, lihat IAM: Memungkinkan pengguna IAM tertentu untuk mengelola grup secara terprogram dan di konsol. Untuk instruksi terperinci dalam membuat kebijakan menggunakan dokumen JSON, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

Mengendalikan akses ke kebijakan

Anda dapat mengontrol cara pengguna menerapkan kebijakan AWS terkelola. Untuk melakukannya, lampirkan kebijakan ini ke semua pengguna Anda. Idealnya, Anda dapat melakukan ini menggunakan sebuah grup pengguna.

Misalnya, Anda dapat membuat kebijakan yang memungkinkan pengguna hanya melampirkan kebijakan IAMUserChangePassworddan PowerUserAccess AWS terkelola ke pengguna, grup pengguna, atau peran IAM baru.

Untuk kebijakan yang dikelola pelanggan, Anda dapat mengontrol siapa yang dapat membuat, memperbarui, dan menghapus kebijakan ini. Anda dapat mengontrol siapa yang dapat melampirkan

dan melepaskan kebijakan ke dan dari entitas utama (grup IAM, pengguna, dan peran). Anda juga dapat mengontrol kebijakan mana yang dapat diberikan atau dilepas pengguna, dan ke dan dari entitas mana.

Misalnya, Anda dapat memberikan izin kepada administrator akun untuk membuat, memperbarui, dan menghapus kebijakan. Kemudian Anda memberikan izin kepada pemimpin tim atau administrator terbatas lainnya untuk memberikan dan melepaskan kebijakan ini ke dan dari entitas prinsipal yang dikelola oleh administrator terbatas.

Untuk informasi selengkapnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat kebijakan IAM yang dapat Anda berikan ke prinsipal, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara memberikan kebijakan IAM pada prinsipal, lihat Menambahkan dan menghapus izin identitas IAM.
- Untuk melihat contoh kebijakan untuk membatasi penggunaan kebijakan yang dikelola, lihat IAM: Membatasi kebijakan terkelola yang dapat diterapkan pada pengguna, grup, atau peran IAM.

Mengontrol izin untuk membuat, memperbarui, dan menghapus kebijakan yang dikelola pelanggan

Anda dapat menggunakan kebijakan IAM untuk mengontrol siapa yang diizinkan untuk membuat, memperbarui, dan menghapus kebijakan yang dikelola pelanggan di Anda Akun AWS. Daftar berikut berisi operasi API yang berkaitan langsung dengan pembuatan, pembaruan, dan penghapusan kebijakan atau versi kebijakan:

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

Operasi API dalam daftar sebelumnya sesuai dengan tindakan yang dapat Anda izinkan atau tolak—yaitu, izin yang dapat Anda berikan—menggunakan kebijakan IAM.

Pertimbangkan contoh kebijakan berikut. Ini memungkinkan pengguna untuk membuat, memperbarui (yaitu, membuat versi kebijakan baru), menghapus, dan menetapkan versi default untuk semua kebijakan yang dikelola pelanggan di Akun AWS. Contoh kebijakan juga mengizinkan pengguna

untuk mencantumkan kebijakan dan mendapatkan kebijakan. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called "Membuat kebijakan menggunakan editor JSON".

Example Contoh kebijakan yang mengizinkan membuat, memperbarui, menghapus, mencantumkan, mendapatkan, dan mengatur versi default untuk semua kebijakan

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Anda dapat membuat kebijakan yang membatasi penggunaan operasi API ini agar hanya memengaruhi kebijakan terkelola yang Anda tentukan. Misalnya, Anda mungkin ingin mengizinkan pengguna untuk mengatur versi default dan menghapus versi kebijakan, tetapi hanya untuk kebijakan tertentu yang dikelola pelanggan. Anda melakukannya dengan menentukan ARN kebijakan dalam elemen `Resource` dari kebijakan yang memberikan izin ini.

Contoh berikut menunjukkan kebijakan yang mengizinkan pengguna menghapus versi kebijakan dan mengatur versi default. Namun tindakan ini hanya diizinkan untuk kebijakan yang dikelola pelanggan yang mencakup jalur /TEAM-A/. ARN kebijakan yang dikelola pelanggan ditentukan dalam elemen `Resource` dari kebijakan. (Dalam contoh ini ARN mencakup jalur dan sebuah wildcard sehingga cocok dengan semua kebijakan yang dikelola pelanggan yang mencakup jalur /TEAM-A/). Untuk

mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

Untuk informasi selengkapnya tentang menggunakan jalur atas nama kebijakan yang dikelola pelanggan, lihat Nama dan jalur yang ramah.

Example Contoh kebijakan yang mengizinkan menghapus versi kebijakan dan mengatur versi default hanya untuk kebijakan tertentu

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:DeletePolicyVersion",
            "iam:SetDefaultPolicyVersion"
        ],
        "Resource": "arn:aws:iam::111122223333:policy/TEAM-A/*"
    }
}
```

Mengontrol izin untuk memberikan dan melepaskan kebijakan yang dikelola

Anda juga dapat menggunakan kebijakan IAM untuk mengizinkan pengguna bekerja dengan hanya kebijakan terkelola khusus saja. Sebagai akibatnya, Anda dapat mengontrol izin mana yang diizinkan untuk diberikan pengguna kepada entitas prinsipal lainnya.

Daftar berikut menunjukkan operasi API yang berkaitan langsung dengan memberikan dan melepaskan kebijakan terkelola ke dan dari entitas prinsipal:

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Anda dapat membuat kebijakan yang membatasi penggunaan operasi API ini agar hanya memengaruhi entitas and/or utama kebijakan terkelola tertentu yang Anda tentukan. Misalnya, Anda mungkin ingin mengizinkan pengguna untuk melampirkan kebijakan terkelola, tetapi hanya kebijakan terkelola yang Anda tentukan. Atau, Anda mungkin ingin mengizinkan pengguna untuk memberikan kebijakan terkelola, tetapi hanya kepada entitas prinsipal yang Anda tentukan.

Contoh kebijakan berikut memungkinkan pengguna untuk melampirkan kebijakan terkelola hanya ke grup dan peran IAM yang menyertakan jalur /TEAM-A/. Grup pengguna dan peran ARNs ditentukan dalam `Resource` elemen kebijakan. (Dalam contoh ini ARNs sertakan jalur dan karakter wildcard dan dengan demikian cocok dengan semua grup dan peran IAM yang menyertakan jalur /TEAM-A/). Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

Example Contoh kebijakan yang mengizinkan memberikan kebijakan terkelola hanya untuk grup pengguna atau peran tertentu

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ]
    }
}
```

Anda selanjutnya dapat membatasi tindakan dalam contoh sebelumnya untuk hanya memengaruhi kebijakan tertentu. Artinya, Anda dapat mengontrol izin yang boleh diberikan pengguna ke entitas prinsipal lainnya—dengan menambahkan ketentuan ke kebijakan.

Dalam contoh berikut, ketentuan ini memastikan bahwa izin `AttachGroupPolicy` dan `AttachRolePolicy` hanya diperbolehkan jika kebijakan yang diberikan sesuai dengan salah

satu kebijakan yang ditentukan. Ketentuan menggunakan `iam:PolicyARN` kunci ketentuan untuk menentukan kebijakan mana yang diizinkan untuk diberikan. Contoh kebijakan berikut berkembang pada contoh sebelumnya. Ini memungkinkan pengguna untuk melampirkan hanya kebijakan terkelola yang menyertakan jalur /TEAM-A/ ke hanya grup dan peran IAM yang menyertakan jalur /TEAM-A/. Untuk mempelajari cara membuat kebijakan dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ],
        "Condition": {
            "ArnLike": {
                "iam:PolicyARN": "arn:aws:iam::111122223333:policy/TEAM-A/*"
            }
        }
    }
}
```

Kebijakan ini menggunakan operator `ArnLike` kondisi, tetapi Anda juga dapat menggunakan operator `ArnEquals` kondisi karena kedua operator kondisi ini berperilaku identik. Untuk informasi lebih lanjut tentang `ArnLike` dan `ArnEquals`, lihat Operator ketentuan Amazon Resource Name (ARN) dalam Jenis Ketentuan bagian dari Referensi Elemen Kebijakan.

Misalnya, Anda dapat membatasi penggunaan tindakan untuk hanya melibatkan kebijakan terkelola yang Anda tentukan. Anda melakukannya dengan menentukan ARN kebijakan dalam elemen `Condition` dari kebijakan yang memberikan izin ini. Misalnya, untuk menentukan ARN kebijakan yang dikelola pelanggan:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Anda juga dapat menentukan ARN kebijakan AWS terkelola dalam elemen kebijakan. `Condition` ARN dari kebijakan AWS terkelola menggunakan alias khusus `aws` dalam kebijakan ARN alih-alih ID akun, seperti dalam contoh ini:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Mengontrol akses ke sumber daya

Anda dapat mengontrol akses ke sumber daya dengan menggunakan kebijakan berbasis sumber daya. Dalam kebijakan berbasis identitas, Anda melampirkan kebijakan ke identitas dan menyebutkan sumber daya apa yang dapat diakses oleh identitas tersebut. Dalam kebijakan berbasis sumber daya, Anda memberikan kebijakan pada sumber daya yang ingin Anda kontrol. Dalam kebijakan, Anda menentukan prinsipal mana yang dapat mengakses sumber daya tersebut. Untuk informasi lebih lanjut tentang kedua jenis kebijakan, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya.

Untuk informasi selengkapnya, lihat sumber daya ini:

- Untuk mempelajari lebih lanjut tentang membuat kebijakan IAM yang dapat Anda berikan ke prinsipal, lihat Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.
- Untuk mempelajari cara memberikan kebijakan IAM pada prinsipal, lihat Menambahkan dan menghapus izin identitas IAM.
- Amazon S3 mendukung penggunaan kebijakan berbasis sumber daya di bucket mereka. Untuk informasi lebih lanjut, lihat Contoh Kebijakan Bucket:

Pembuat Sumber Daya Tidak Secara Otomatis Memiliki Izin

Jika Anda masuk menggunakan Pengguna root akun AWS kredensyal, Anda memiliki izin untuk melakukan tindakan apa pun pada sumber daya milik akun. Namun, ini tidak benar untuk pengguna IAM. Pengguna IAM dapat diberikan akses untuk membuat sumber daya, tetapi izin pengguna, bahkan untuk sumber daya tersebut, terbatas untuk apa yang telah diberikan secara jelas. Ini berarti bahwa hanya karena Anda membuat sumber daya, seperti peran IAM, Anda tidak secara otomatis

memiliki izin untuk mengubah atau menghapus peran tersebut. Sebagai tambahan, izin Anda dapat dicabut setiap saat oleh pemilik akun atau pengguna lain yang telah diberi akses untuk mengelola izin Anda.

Mengontrol akses ke prinsipal dalam akun tertentu

Anda dapat memberi izin pengguna IAM secara langsung di akses akun Anda sendiri ke sumber daya Anda. Jika pengguna dari akun lain memerlukan akses ke sumber daya Anda, Anda dapat membuat peran IAM. Peran adalah sebuah entitas yang mencakup izin tetapi tidak terkait dengan pengguna tertentu. Pengguna dari akun lain kemudian dapat menggunakan peran dan mengakses sumber daya sesuai dengan izin yang telah Anda tetapkan untuk peran tersebut. Untuk informasi selengkapnya, lihat Akses untuk pengguna IAM di yang lain Akun AWS yang Anda miliki.

> **Note**
>
> Beberapa layanan mendukung kebijakan berbasis sumber daya sebagaimana dijelaskan dalam Identity-based kebijakan dan kebijakan berbasis sumber daya (seperti Amazon S3, Amazon SNS, dan Amazon SQS). Untuk layanan tersebut, alternatif untuk menggunakan peran adalah dengan memberikan kebijakan ke sumber daya (bucket, topik, atau antrean) yang ingin Anda bagikan. Kebijakan berbasis sumber daya dapat menentukan AWS akun yang memiliki izin untuk mengakses sumber daya.

Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag

Gunakan informasi di bagian berikut untuk mengontrol siapa yang dapat mengakses pengguna dan peran IAM Anda serta sumber daya apa yang dapat diakses pengguna dan peran Anda. Untuk informasi lebih umum dan contoh pengendalian akses ke AWS sumber daya lain, termasuk sumber daya IAM lainnya, lihatTag untuk AWS Identity and Access Management sumber daya.

> **Note**
>
> Untuk detail tentang sensitivitas huruf besar untuk kunci tag dan nilai kunci tag, lihatCase sensitivity.

Tag dapat diberikan ke sumber daya IAM, melewati permintaan, atau diberikan pada prinsipal yang membuat permintaan. Pengguna atau peran IAM dapat berupa sumber daya dan prinsipal. Misalnya,

Anda dapat menulis sebuah kebijakan yang mengizinkan pengguna untuk mencantumkan grup untuk pengguna. Operasi ini hanya diizinkan jika pengguna yang mengajukan permintaan (principal) memiliki tanda `project=blue` yang sama sebagai pengguna yang mereka coba lihat. Dalam contoh ini, pengguna dapat melihat keanggotaan kelompok untuk pengguna mana pun, termasuk mereka sendiri, selama mereka bekerja pada proyek yang sama.

tuanUntuk mengontrol akses berdasarkan tag, Anda memberikan informasi tag di elemen ketentuan dari kebijakan. Saat Anda membuat kebijakan IAM, Anda dapat menggunakan tag IAM dan kunci ketentuan tag terkait untuk mengontrol akses ke salah satu hal berikut:

- Sumber Daya – Mengontrol akses ke sumber daya pengguna atau peran berdasarkan tag mereka. Untuk melakukan ini, gunakan kunci *key-name* kondisi aws:ResourceTag/untuk menentukan pasangan nilai kunci tag mana yang harus dilampirkan ke sumber daya. Untuk informasi selengkapnya, lihat Mengontrol akses ke AWS sumber daya.
- Permintaan – Mengontrol tag yang dapat diberikan dalam permintaan IAM. Untuk melakukan ini, gunakan kunci *key-name* kondisi aws:RequestTag/untuk menentukan tag apa yang dapat ditambahkan, diubah, atau dihapus dari pengguna atau peran IAM. Kunci ini digunakan dengan cara yang sama untuk sumber daya IAM dan sumber AWS daya lainnya. Untuk informasi selengkapnya, lihat Mengontrol akses selama AWS permintaan.
- Prinsipal – Mengontrol apa yang boleh dilakukan oleh orang yang membuat permintaan (prinsipal) berdasarkan tag yang dilampirkan ke pengguna atau peran IAM orang tersebut. Untuk melakukan ini, gunakan kunci *key-name* kondisi aws:PrincipalTag/untuk menentukan tag apa yang harus dilampirkan ke pengguna atau peran IAM sebelum permintaan diizinkan.
- Setiap bagian dari proses otorisasi — Gunakan aws: TagKeys condition key untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan atau oleh prinsipal. Dalam hal ini, nilai kunci tidak menjadi masalah. Kunci ini berperilaku sama untuk IAM dan layanan lainnya AWS . Namun, ketika Anda menandai pengguna di IAM, ini juga mengontrol apakah prinsipal dapat membuat permintaan ke layanan apa pun. Untuk informasi selengkapnya, lihat Mengontrol akses berdasarkan kunci tanda.

Anda dapat membuat kebijakan IAM menggunakan editor visual, dengan menggunakan JSON, atau dengan mengimpor kebijakan terkelola yang ada. Lihat perinciannya di Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.

> **Note**
>
> Anda juga dapat meneruskan tag sesi saat Anda mengambil peran IAM atau mengfederasi pengguna. Ini hanya berlaku untuk durasi sesi.

Mengontrol akses untuk prinsipal IAM

Anda dapat mengontrol apa yang diperbolehkan untuk penanggung jawab berdasarkan tanda yang terpasang pada identitas orang tersebut.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan setiap pengguna di akun ini untuk melihat keanggotaan grup untuk setiap pengguna, termasuk mereka sendiri, selama mereka mengerjakan proyek yang sama. Operasi ini hanya diperbolehkan jika tag sumber daya pengguna dan tag prinsipal memiliki nilai yang sama untuk kunci tag`project`. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

Mengontrol akses berdasarkan kunci tag

Anda dapat menggunakan tag dalam kebijakan IAM Anda untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan atau oleh prinsipal.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan hanya menghapus tag dengan `temporary` kunci dari pengguna. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys":
 ["temporary"]}}
    }]
}
```

Mengontrol akses ke AWS sumber daya menggunakan tag

Anda dapat menggunakan tag untuk mengontrol akses ke AWS sumber daya yang mendukung penandaan, termasuk sumber daya IAM. Anda dapat menandai pengguna dan peran IAM untuk mengontrol apa yang dapat mereka akses. Untuk mempelajari cara menandai pengguna dan peran IAM, lihat Tag untuk AWS Identity and Access Management sumber daya. Selain itu, Anda dapat mengontrol akses ke sumber daya IAM berikut: kebijakan yang dikelola pelanggan, penyedia identitas IAM, profil instans, sertifikat server, dan perangkat MFA virtual. Untuk melihat tutorial membuat dan menguji kebijakan yang mengizinkan peran IAM dengan tag prinsipal mengakses sumber daya dengan tag yang cocok, lihat Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag. Gunakan informasi di bagian berikut untuk mengontrol akses ke AWS sumber daya lain, termasuk sumber daya IAM, tanpa menandai pengguna atau peran IAM.

Sebelum Anda menggunakan tag untuk mengontrol akses ke AWS sumber daya Anda, Anda harus memahami bagaimana AWS memberikan akses. AWS terdiri dari koleksi sumber daya. Instans

Amazon EC2 adalah sebuah sumber daya. Bucket Amazon S3 adalah sebuah sumber daya. Anda dapat menggunakan AWS API, file AWS CLI, atau Konsol Manajemen AWS untuk melakukan operasi, seperti membuat bucket di Amazon S3. Saat Anda melakukannya, Anda mengirimkan sebuah permintaan untuk operasi tersebut. Permintaan Anda menentukan tindakan, sumber daya, sebuah entitas prinsipal (pengguna atau peran), sebuah akun prinsipal, dan informasi permintaan yang diperlukan. Semua informasi ini memberikan konteks.

AWS kemudian memeriksa bahwa Anda (entitas utama) diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk melakukan tindakan yang ditentukan pada sumber daya yang ditentukan. Selama otorisasi, AWS periksa semua kebijakan yang berlaku untuk konteks permintaan Anda. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON dan menentukan izin untuk entitas utama. Untuk informasi selengkapnya tentang tipe dan penggunaan kebijakan, lihat Kebijakan dan izin di AWS Identity and Access Management.

AWS mengotorisasi permintaan hanya jika setiap bagian dari permintaan Anda diizinkan oleh kebijakan. Untuk melihat diagram dan lebih mempelajari tentang infrastruktur IAM, lihat Cara kerja IAM. Untuk detail tentang bagaimana IAM menentukan apakah permintaan diizinkan, lihat Logika evaluasi kebijakan.

Tag adalah pertimbangan lain dalam proses ini karena tag dapat dilampirkan ke sumber daya atau diteruskan dalam permintaan ke layanan yang mendukung penandaan. Untuk mengontrol akses berdasarkan tanda, Anda memberikan informasi tanda di elemen ketentuan dari kebijakan. Untuk mempelajari apakah suatu AWS layanan mendukung pengendalian akses menggunakan tag, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom ABAC. Pilih nama layanan untuk melihat dokumentasi otorisasi dan kontrol akses untuk layanan tersebut.

Anda kemudian dapat membuat kebijakan IAM yang mengizinkan atau menolak akses ke sumber daya berdasarkan tag sumber daya tersebut. Dalam kebijakan tersebut, Anda dapat menggunakan kunci ketentuan tag untuk mengontrol akses ke salah satu hal berikut:

- Sumber Daya — Kontrol akses ke sumber daya AWS layanan berdasarkan tag pada sumber daya tersebut. Untuk melakukan ini, gunakan kunci *key-name* kondisi aws:ResourceTag/untuk menentukan apakah akan mengizinkan akses ke sumber daya berdasarkan tag yang dilampirkan ke sumber daya.
- Permintaan – Mengontrol tag apa yang dapat diteruskan dalam permintaan. Untuk melakukan ini, gunakan kunci *key-name* kondisi aws:RequestTag/untuk menentukan pasangan nilai kunci tag apa yang dapat diteruskan dalam permintaan untuk menandai sumber daya. AWS

- Setiap bagian dari proses otorisasi — Gunakan aws: TagKeys condition key untuk mengontrol apakah kunci tag tertentu dapat berada dalam permintaan.

Anda dapat membuat kebijakan IAM secara visual, menggunakan JSON, atau dengan mengimpor kebijakan terkelola yang sudah ada. Lihat perinciannya di Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan.

> **Note**
>
> Beberapa layanan memungkinkan pengguna untuk menentukan tag ketika mereka membuat sumber daya jika mereka memiliki izin untuk menggunakan tindakan yang membuat sumber daya.

Mengontrol akses ke AWS sumber daya

Anda dapat menggunakan kondisi dalam kebijakan IAM untuk mengontrol akses ke AWS sumber daya berdasarkan tag pada sumber daya tersebut. Anda dapat melakukannya menggunakan kunci syarat `aws:ResourceTag/tag-key`, atau kunci syarat layanan khusus. Beberapa layanan hanya mendukung versi khusus layanan kunci ini dan bukan versi global.

> **Warning**
>
> Jangan mencoba mengontrol siapa yang dapat melewati peran dengan menandai peran dan kemudian menggunakan kunci `ResourceTag` kondisi dalam kebijakan dengan `iam:PassRole` tindakan tersebut. Pendekatan ini tidak memiliki hasil yang dapat diandalkan. Untuk informasi lebih lanjut tentang izin yang diperlukan untuk memberikan sebuah peran ke sebuah layanan, lihat Memberikan izin pengguna untuk meneruskan peran ke AWS layanan.

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instans Amazon EC2. Operasi ini hanya diperbolehkan jika tag instance `Owner` memiliki nilai nama pengguna. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Anda dapat melampirkan kebijakan ini ke pengguna IAM di akun Anda. Jika pengguna bernama `richard` mencoba memulai instans Amazon EC2, instans harus diberi tag `Owner=richard` atau. `owner=richard` Atau, aksesnya akan ditolak. Kunci tanda `Owner` cocok dengan kedua `Owner` dan `owner` karena nama kunci ketentuan tidak terpengaruh huruf besar/kecil. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM JSON: Kondisi.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menggunakan tag `team` utama di ARN sumber daya. Kebijakan memberikan izin untuk menghapus antrean Amazon Simple Queue Service, tetapi hanya jika nama antrian dimulai dengan nama tim yang diikuti. `-queue` Misalnya, `qa-queue` if `qa` adalah nama tim untuk tag `team` utama.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
```

```
        "Resource": "arn:aws:sqs:us-east-2:111122223333:${aws:PrincipalTag/team}-
queue"
      }
}
```

Mengontrol akses selama AWS permintaan

Anda dapat menggunakan kondisi dalam kebijakan IAM untuk mengontrol pasangan nilai kunci tag apa yang dapat diteruskan dalam permintaan yang menerapkan tag ke sumber daya. AWS

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan `CreateTags` tindakan Amazon EC2 untuk melampirkan tag ke instance. Anda dapat memberikan tanda hanya jika tanda berisi kunci `environment` dan `preprod` atau nilai `production`. Jika anda mau, Anda bisa menggunakan pengubah `ForAllValues` dengan kunci ketentuan `aws:TagKeys` untuk menunjukkan bahwa hanya kunci `environment` yang diperbolehkan dalam permintaan. Ini menghentikan pengguna untuk memasukkan kunci lain, seperti tidak sengaja menggunakan `Environment` daripada `environment`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Mengontrol akses berdasarkan kunci tanda

Anda dapat menggunakan kondisi dalam kebijakan IAM untuk mengontrol apakah kunci tag tertentu dapat digunakan dalam permintaan.

Kami menyarankan bahwa ketika Anda menggunakan kebijakan untuk mengontrol akses menggunakan tag, Anda menggunakan tombol `aws:TagKeys` kondisi. AWS layanan yang mendukung tag memungkinkan Anda membuat beberapa nama kunci tag yang hanya berbeda berdasarkan kasus, seperti menandai instans `stack=production` Amazon EC2 dengan dan. `Stack=test` Nama-nama kunci tidak peka huruf dalam syarat kebijakan. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya. Untuk mencegah tag duplikat dengan kunci yang hanya bervariasi menurut kasus, gunakan `aws:TagKeys` kondisi untuk menentukan kunci tag yang dapat diterapkan pengguna Anda, atau gunakan kebijakan tag, yang AWS Organizations tersedia. Untuk informasi selengkapnya, lihat Kebijakan Tag di Panduan AWS Organizations Pengguna.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pembuatan dan penandaan rahasia Secrets Manager, tetapi hanya dengan kunci tag atau. `environment` `cost-center` `Null`Kondisi memastikan bahwa kondisi mengevaluasi `false` jika tidak ada tag dalam permintaan.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
```

```
}
```

Akses sumber daya lintas akun di IAM

Untuk beberapa AWS layanan, Anda dapat memberikan akses lintas akun ke sumber daya menggunakan IAM. Untuk melakukan ini, Anda dapat melampirkan kebijakan sumber daya langsung ke sumber daya yang ingin Anda bagikan, atau menggunakan peran sebagai proxy.

Untuk berbagi sumber daya secara langsung, sumber daya yang ingin Anda bagikan harus mendukung kebijakan berbasis sumber daya. Tidak seperti kebijakan berbasis identitas untuk peran, kebijakan berbasis sumber daya menentukan siapa (kepala sekolah mana) yang dapat mengakses sumber daya tersebut.

Gunakan peran sebagai proxy saat Anda ingin mengakses sumber daya di akun lain yang tidak mendukung kebijakan berbasis sumber daya.

Untuk detail tentang perbedaan antara jenis kebijakan ini, lihatIdentity-based kebijakan dan kebijakan berbasis sumber daya.

> **Note**
>
> Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dengan satu partisi. Misalnya, Anda memiliki akun di AS Barat (N. California) di `aws` partisi standar. Anda juga memiliki akun di China di `aws-cn` partisi. Anda tidak dapat menggunakan kebijakan berbasis sumber daya di akun Anda di China untuk mengizinkan akses bagi pengguna di akun standar AWS Anda.

Cross-account akses menggunakan peran

Tidak semua AWS layanan mendukung kebijakan berbasis sumber daya. Untuk layanan ini, Anda dapat menggunakan peran IAM lintas akun untuk memusatkan manajemen izin saat menyediakan akses lintas akun ke beberapa layanan. Peran IAM lintas akun adalah peran IAM yang mencakup kebijakan kepercayaan yang memungkinkan kepala sekolah IAM di AWS akun lain untuk mengambil peran tersebut. Sederhananya, Anda dapat membuat peran dalam satu AWS akun yang mendelegasikan izin tertentu ke AWS akun lain.

Untuk informasi tentang melampirkan kebijakan ke identitas IAM, lihat Kelola kebijakan IAM.

> **Note**
>
> Ketika kepala sekolah beralih ke peran untuk sementara menggunakan izin peran, mereka menyerahkan izin aslinya dan mengambil izin yang ditetapkan untuk peran yang telah mereka ambil.

Mari kita lihat keseluruhan proses yang berlaku untuk perangkat lunak Mitra APN yang perlu mengakses akun pelanggan.

1. Pelanggan membuat peran IAM di akun mereka sendiri dengan kebijakan yang memungkinkan akses sumber daya Amazon S3 yang dibutuhkan mitra APN. Dalam contoh ini, nama peran adalah`APNPartner`.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "s3:*",
               "Resource": [
                   "arn:aws:s3:::bucket-name"
               ]
           }
       ]
   }
   ```

2. Kemudian, pelanggan menentukan bahwa peran dapat diasumsikan oleh AWS akun mitra dengan memberikan Akun AWS ID Mitra APN dalam kebijakan kepercayaan untuk `APNPartner` peran tersebut.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
   ```

```
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Pelanggan memberikan Nama Sumber Daya Amazon (ARN) dari peran tersebut kepada mitra APN. ARN adalah nama peran yang sepenuhnya memenuhi syarat.

```
arn:aws:iam::Customer-Account-ID:role/APNPartner
```

> **Note**
>
> Sebaiknya gunakan ID eksternal dalam situasi multi-penyewa. Lihat perinciannya di Akses ke Akun AWS dimiliki oleh pihak ketiga.

4. Ketika perangkat lunak Mitra APN perlu mengakses akun pelanggan, perangkat lunak memanggil AssumeRole API AWS Security Token Service dengan ARN peran di akun pelanggan. STS mengembalikan AWS kredensi sementara yang memungkinkan perangkat lunak melakukan pekerjaannya.

Untuk contoh lain pemberian akses lintas akun menggunakan peran, lihatAkses untuk pengguna IAM di yang lain Akun AWS yang Anda miliki. Anda juga dapat mengikutiTutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM.

Cross-account akses menggunakan kebijakan berbasis sumber daya

Ketika akun mengakses sumber daya melalui akun lain menggunakan kebijakan berbasis sumber daya, prinsipal masih bekerja di akun tepercaya dan tidak harus menyerahkan izin mereka untuk menerima izin peran. Dengan kata lain, kepala sekolah terus memiliki akses ke sumber daya di akun tepercaya sambil memiliki akses ke sumber daya di akun yang dipercaya. Ini berguna untuk tugas seperti menyalin informasi ke atau dari sumber daya yang dibagikan di akun lain.

Prinsipal yang dapat Anda tentukan dalam kebijakan berbasis sumber daya termasuk akun, pengguna IAM, prinsipal pengguna federasi, prinsipal AWS STS federasi SAML, prinsipal federasi

OIDC, peran IAM, sesi peran yang diasumsikan, atau layanan. AWS Untuk informasi selengkapnya, lihat Menentukan prinsipal.

Untuk mempelajari apakah kepala sekolah di akun di luar zona kepercayaan Anda (organisasi atau akun tepercaya) memiliki akses untuk mengambil peran Anda, lihat Mengi dentifikasi sumber daya yang dibagikan dengan entitas eksternal.

Daftar berikut mencakup beberapa AWS layanan yang mendukung kebijakan berbasis sumber daya. Untuk daftar lengkap peningkatan jumlah AWS layanan yang mendukung melampirkan kebijakan izin ke sumber daya alih-alih prinsipal, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Berbasis Sumber Daya.

- Bucket Amazon S3 — Kebijakan dilampirkan ke bucket, tetapi kebijakan mengontrol akses ke bucket dan objek di dalamnya. Untuk informasi selengkapnya, lihat Kebijakan Bucket untuk Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Dalam beberapa kasus, yang terbaik mungkin adalah menggunakan peran untuk akses lintas akun ke Amazon S3 Untuk informasi selengkapnya, lihat contoh panduan di Panduan Pengguna Layanan Peny impanan Sederhana Amazon.
- Topik Amazon Simple Notification Service (Amazon SNS) — Untuk informasi selengkapnya, lihat Contoh kasus untuk kontrol akses Amazon SNS di Panduan Pengembang Layanan Pem beritahuan Sederhana Amazon.
- Antrian Amazon Simple Queue Service (Amazon SQS) — Untuk informasi selengkapnya, lihat Lampiran: Bahasa Kebijakan Akses di Panduan Pengembang Layanan Antrian Sederhana Amazon.

Resource-based Kebijakan untuk mendelegasikan AWS izin

Jika sumber daya memberikan izin kepada prinsipal di akun Anda, maka Anda dapat mendelegasikan izin tersebut kepada identitas IAM tertentu. Identitas adalah pengguna, kelompok pengguna, atau peran dalam akun Anda. Anda mendelegasikan izin dengan melampirkan kebijakan ke identitas. Anda dapat memberikan izin maksimum yang diizinkan oleh akun pemilik sumber daya.

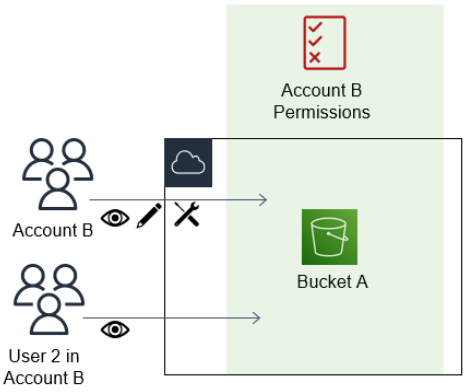
> **Important**
>
> Dalam akses lintas akun, prinsipal membutuhkan kebijakan identitas dan kebijakan berbasis sumber daya. `Allow`

Asumsikan bahwa kebijakan berbasis sumber daya memberi semua prinsipal di akun Anda akses administratif penuh ke sumber daya. Kemudian Anda dapat mendelegasikan akses penuh, akses baca-saja, atau akses parsional lainnya ke kepala sekolah di akun Anda. AWS Alternatifnya, jika kebijakan berbasis sumber daya hanya memberikan izin daftar, maka Anda hanya dapat mendelegasikan akses daftar. Jika Anda mencoba untuk mendelegasikan lebih banyak izin dari yang dimiliki akun Anda, prinsipal Anda masih hanya akan memiliki akses daftar.

Untuk informasi selengkapnya tentang bagaimana keputusan ini dibuat, lihat Men entukan apakah permintaan diizinkan atau ditolak dalam akun.

> **Note**
>
> Peran IAM dan kebijakan berbasis sumber daya mendelegasikan akses ke seluruh akun hanya dengan satu partisi. Misalnya, Anda tidak dapat menambahkan akses lintas akun antara akun dalam partisi `aws` standar dan akun dalam partisi `aws-cn`.

Sebagai contoh, anggaplah bahwa Anda mengelola `AccountA` dan `AccountB`. Di AccountA, Anda memiliki bucket Amazon S3 bernama`BucketA`.



1. Anda melampirkan kebijakan berbasis sumber daya `BucketA` yang memungkinkan semua kepala sekolah di AccounTB akses penuh ke objek di bucket Anda. Mereka dapat membuat, membaca, atau menghapus objek apa pun dalam bucket tersebut.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

AccountA memberi AccountB akses penuh ke BucketA dengan menamai accountB sebagai prinsipal dalam kebijakan berbasis sumber daya. Akibatnya, AccountB diberi wewenang untuk melakukan tindakan apa pun pada BucketA, dan administrator AccountB dapat mendelegasikan akses ke penggunanya di AccountB.

Pengguna root AccountB memiliki semua izin yang diberikan ke akun. Oleh karena itu, pengguna root memiliki akses penuh ke BucketA.

2. Di AccountB, lampirkan kebijakan ke pengguna IAM bernama User2. Kebijakan itu memungkinkan pengguna akses read-only ke objek di BucketA. Itu berarti bahwa User2 dapat melihat objek, tetapi tidak membuat, mengedit, atau menghapusnya.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

Tingkat akses maksimum yang dapat didelegasikan AccountB adalah tingkat akses yang diberikan ke akun. Dalam hal ini, kebijakan berbasis sumber daya memberikan akses penuh ke AccountB, tetapi User2 hanya diberikan akses baca-saja.

Administrator AccountB tidak memberikan akses ke User1. Secara default, pengguna tidak memiliki izin apa pun kecuali yang diberikan secara eksplisit, sehingga User1 tidak memiliki akses ke BucketA.

IAM mengevaluasi izin prinsipal pada saat prinsipal mengajukan permintaan. Jika Anda menggunakan wildcard (*) untuk memberi pengguna akses penuh ke sumber daya Anda, kepala sekolah dapat mengakses sumber daya apa pun yang dapat diakses AWS akun Anda. Hal ini berlaku bahkan untuk sumber daya yang Anda tambahkan atau Anda memiliki akses kepadanya setelah membuat kebijakan pengguna.

Dalam contoh sebelumnya, jika accountB telah melampirkan kebijakan ke Pengguna2 yang mengizinkan akses penuh ke semua sumber daya di semua akun, Pengguna2 akan secara otomatis memiliki akses ke sumber daya apa pun yang dapat diakses AccountB. Ini termasuk akses BucketA dan akses ke sumber daya lain yang diberikan oleh kebijakan berbasis sumber daya di AccountA.

Untuk informasi selengkapnya tentang penggunaan peran yang kompleks, seperti memberikan akses ke aplikasi dan layanan, lihatSkenario umum untuk peran IAM.

> **Important**
>
> Berikan akses hanya kepada entitas yang Anda percayai, dan berikan tingkat akses minimum yang diperlukan. Setiap kali entitas tepercaya adalah AWS akun lain, prinsipal IAM apa pun dapat diberikan akses ke sumber daya Anda. Akun tepercaya AWS hanya dapat mendelegasikan akses sejauh telah diberikan akses; akun tepercaya tidak dapat mendelegasikan lebih banyak akses daripada yang diberikan akun itu sendiri.

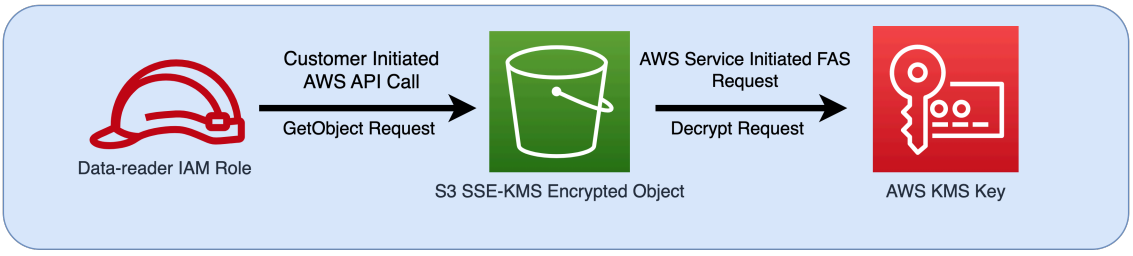
Untuk informasi tentang izin, kebijakan, dan bahasa kebijakan izin yang Anda gunakan untuk menyusun kebijakan, lihat Manajemen akses untuk AWS sumber daya.

Teruskan sesi akses

Sesi akses maju (FAS) adalah teknologi IAM yang digunakan oleh AWS layanan untuk meneruskan identitas, izin, dan atribut sesi Anda ketika AWS layanan membuat permintaan atas nama Anda. FAS menggunakan izin identitas memanggil AWS layanan, dikombinasikan dengan identitas AWS layanan untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ke AWS layanan atas nama kepala sekolah IAM setelah layanan menerima permintaan yang memerlukan interaksi dengan AWS layanan atau sumber daya lain untuk diselesaikan. Ketika permintaan FAS dibuat:

- Layanan yang menerima permintaan awal dari prinsipal IAM memeriksa izin prinsipal IAM.
- Layanan yang menerima permintaan FAS berikutnya juga memeriksa izin dari prinsipal IAM yang sama.

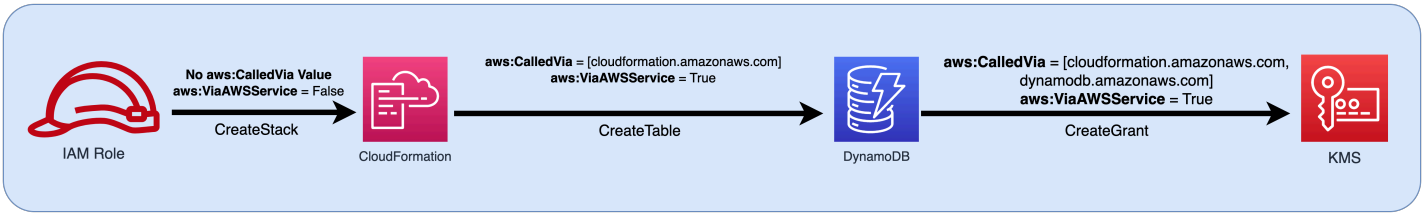
Misalnya, FAS digunakan oleh Amazon S3 untuk melakukan panggilan AWS Key Management Service untuk mendekripsi objek SSE-KMS ketika digunakan untuk mengenkripsinya. Saat mengunduh objek SSE-KMS terenkripsi, peran bernama data-reader memanggil objek GetObject terhadap Amazon S3, dan tidak memanggil AWS KMS secara langsung. Setelah menerima GetObject permintaan dan mengotorisasi pembaca data, Amazon S3 kemudian membuat permintaan FAS untuk AWS KMS mendekripsi objek Amazon S3. Ketika KMS menerima permintaan FAS, ia memeriksa izin peran dan hanya mengotorisasi permintaan dekripsi jika pembaca data memiliki izin yang benar pada kunci KMS. Permintaan ke Amazon S3 dan AWS KMS diotorisasi menggunakan izin peran dan hanya berhasil jika pembaca data memiliki izin ke objek Amazon S3 dan kunci AWS KMS.



> **Note**
>
> Permintaan FAS tambahan dapat dibuat oleh layanan yang telah menerima permintaan FAS. Dalam kasus seperti itu, kepala sekolah yang meminta harus memiliki izin untuk semua layanan yang dipanggil oleh FAS.

Permintaan FAS dan ketentuan kebijakan IAM

Ketika permintaan FAS dibuataws: CalledVia,aws: CalledViaFirst,, dan kunci aw: CalledViaLast kondisi diisi dengan prinsipal layanan dari layanan yang memulai panggilan FAS. Nilai kunci aws:ViaAWSService kondisi disetel ke `true` setiap kali permintaan FAS dibuat. Dalam diagram berikut, permintaan untuk CloudFormation secara langsung tidak memiliki kunci `aws:CalledVia` atau `aws:ViaAWSService` kondisi yang ditetapkan. Ketika CloudFormation dan DynamoDB membuat permintaan FAS hilir atas nama peran, nilai untuk kunci kondisi ini diisi.



Untuk mengizinkan permintaan FAS dibuat ketika permintaan tersebut ditolak oleh pernyataan kebijakan Tolak dengan kunci kondisi pengujian alamat IP Sumber atau VPC Sumber, Anda harus menggunakan kunci kondisi untuk memberikan pengecualian untuk permintaan FAS dalam kebijakan Tolak Anda. Ini dapat dilakukan untuk semua permintaan FAS dengan menggunakan kunci `aws:ViaAWSService` kondisi. Untuk mengizinkan hanya AWS layanan tertentu untuk membuat permintaan FAS, gunakan`aws:CalledVia`.

> **Important**
>
> Ketika permintaan FAS dibuat setelah permintaan awal dibuat melalui titik akhir VPC, nilai kunci kondisi untuk`aws: SourceVpce`,`aws: SourceVpc`, dan `aw: VpcSourceIp` dari permintaan awal tidak digunakan dalam permintaan FAS. Saat menulis kebijakan menggunakan `aws:VpcSourceIp` atau memberikan `aws:SourceVpce` akses secara kondisional, Anda juga harus menggunakan `aws:ViaAWSService` atau mengi `aws:CalledVia` zinkan permintaan FAS. Ketika permintaan FAS dibuat setelah permintaan awal diterima oleh titik akhir AWS layanan publik, permintaan FAS berikutnya akan dibuat dengan nilai kunci `aws:SourceIP` kondisi yang sama.

Contoh: Izinkan akses Amazon S3 dari VPC atau dengan FAS

Dalam contoh kebijakan IAM berikut, permintaan Amazon S3 GetObject dan Athena hanya diperbolehkan jika mereka berasal dari titik akhir VPC yang dilampirkan*example_vpc*, atau jika permintaan tersebut adalah permintaan FAS yang dibuat oleh Athena.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
          "vpc-111bbb22"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Untuk contoh tambahan menggunakan kunci kondisi untuk mengizinkan akses FAS, lihat contoh repo kebijakan perimeter data.

Contoh kebijakan berbasis identitas IAM

Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika kepala sekolah IAM (pengguna atau peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON yang dilampirkan ke identitas IAM (pengguna, grup pengguna, atau peran). Kebijakan berbasis identitas dapat mencakup kebijakan terkelola AWS , kebijakan terkelola pelanggan, dan kebijakan inline. Untuk mempelajari cara membuat kebijakan IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat the section called “Membuat kebijakan menggunakan editor JSON”.

Secara default semua permintaan ditolak, jadi Anda harus memberikan akses ke layanan, tindakan, dan sumber daya yang Anda inginkan bagi identitas untuk mengakses. Jika Anda juga ingin mengizinkan akses untuk menyelesaikan tindakan yang ditentukan di konsol IAM, Anda perlu memberikan izin tambahan.

Pustaka kebijakan berikut dapat membantu Anda menentukan izin untuk identitas IAM Anda. Setelah anda menemukan kebijakan yang Anda perlukan, pilih lihat kebijakan ini untuk melihat JSON untuk kebijakan tersebut. Anda dapat menggunakan dokumen kebijakan JSON sebagai contoh untuk kebijakan Anda sendiri.

> **Note**
>
> Jika Anda ingin memberikan sebuah kebijakan untuk disertakan dalam panduan referensi ini, gunakan tombol Umpan Balik di bagian bawah halaman ini.

Contoh kebijakan: AWS

- Mengizinkan akses selama rentang tanggal tertentu. (Lihat kebijakan ini.)
- Memungkinkan mengaktifkan dan menonaktifkan Wilayah AWS . (Lihat kebijakan ini.)
- Memungkinkan pengguna yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Memungkinkan akses khusus saat menggunakan MFA selama rentang tanggal tertentu. (Lihat kebijakan ini.)

- Memungkinkan pengguna mengelola kredensialnya sendiri di halaman Kredensial Keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola kata sandi mereka sendiri di halaman Kredensi Keamanan. (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan. (Lihat kebijakan ini.)
- Menolak akses AWS berdasarkan Wilayah yang diminta. (Lihat kebijakan ini.)
- Menolak akses AWS berdasarkan alamat IP sumber. (Lihat kebijakan ini.)

Contoh kebijakan: AWS Data Exchange

- Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali. AWS Data Exchange(Lihat kebijakan ini.)

Contoh kebijakan: AWS Data Pipeline

- Menolak akses ke saluran pipa yang tidak dibuat pengguna (Lihat kebijakan ini.)

Contoh kebijakan: Amazon DynamoDB

- Mengizinkan akses ke tabel Amazon DynamoDB tertentu (Lihat kebijakan ini.)
- Mengizinkan akses ke atribut Amazon DynamoDB tertentu (Lihat kebijakan ini.)
- Mengizinkan akses tingkat item ke Amazon DynamoDB berdasarkan ID Amazon Cognito (Lihat kebijakan ini.)

Contoh kebijakan: Amazon EC2

- Mengizinkan melampirkan atau melepaskan volume Amazon EBS ke EC2 instans Amazon berdasarkan tag (Lihat kebijakan ini.)
- Memungkinkan peluncuran EC2 instans Amazon di subnet tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)

- Memungkinkan mengelola grup EC2 keamanan Amazon yang terkait dengan VPC tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan EC2 instans Amazon yang ditandai pengguna, secara terprogram, dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan EC2 instans Amazon berdasarkan sumber daya dan tag utama, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan memulai atau menghentikan EC2 instans Amazon saat sumber daya dan tag utama cocok (Lihat kebijakan ini.)
- Memungkinkan EC2 akses Amazon penuh dalam Wilayah tertentu, secara terprogram dan di konsol. (Lihat kebijakan ini.)
- Memungkinkan memulai atau menghentikan EC2 instans Amazon tertentu dan memodifikasi grup keamanan tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Menolak akses ke EC2 operasi Amazon tertentu tanpa MFA (Lihat kebijakan ini.)
- Membatasi penghentian EC2 instans Amazon ke rentang alamat IP tertentu (Lihat kebijakan ini.)

Contoh kebijakan: AWS Identity and Access Management (IAM)

- Mengizinkan akses ke API simulator kebijakan (Lihat kebijakan ini.)
- Mengizinkan akses ke konsol simulator kebijakan (Lihat kebijakan ini.)
- Mengizinkan untuk memberikan peran apa pun yang memiliki tanda tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan dan menolak akses ke beberapa layanan, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penambahan tag spesifik ke pengguna IAM dengan sebuah tag spesifik yang berbeda, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penambahan tag spesifik ke setiap pengguna IAM atau peran, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pembuatan pengguna baru hanya dengan tag spesifik (Lihat kebijakan ini.)
- Mengizinkan menghasilkan dan mengambil laporan kredensial IAM (Lihat kebijakan ini.)
- Mengizinkan pengelolaan keanggotaan grup, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pengelolaan tag tertentu (Lihat kebijakan ini.)
- Mengizinkan untuk memberikan sebuah peran IAM ke layanan tertentu (Lihat kebijakan ini.)
- Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan (Lihat kebijakan ini.)

- Mengizinkan akses hanya-baca ke konsol IAM (Lihat kebijakan ini.)
- Mengizinkan pengguna spesifik untuk mengelola sebuah grup, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pengaturan persyaratan kata sandi akun, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan penggunaan API simulator kebijakan untuk pengguna dengan jalan tertentu (Lihat kebijakan ini.)
- Mengizinkan penggunaan konsol simulator kebijakan untuk pengguna dengan jalan tertentu (Lihat kebijakan ini.)
- Mengizinkan pengguna IAM untuk mengatur sendiri perangkat MFA. (Lihat kebijakan ini.)
- Memungkinkan pengguna IAM untuk mengatur kredensialnya sendiri, secara terprogram dan di konsol. (Lihat kebijakan ini.)
- Memungkinkan layanan tampilan informasi yang terakhir diakses untuk AWS Organizations kebijakan di konsol IAM. (Lihat kebijakan ini.)
- Batasi kebijakan terkelola yang dapat diterapkan ke pengguna, kelompok atau peran IAM (Lihat kebijakan ini.)
- Mengizinkan akses ke kebijakan IAM hanya di akun Anda (Lihat kebijakan ini.)

Contoh kebijakan: AWS Lambda

- Mengizinkan AWS Lambda fungsi mengakses tabel Amazon DynamoDB (Lihat kebijakan ini.)

Contoh kebijakan: Amazon RDS

- Mengizinkan akses basis data RDS sepenuhnya di dalam Wilayah tertentu (Lihat kebijakan ini.)
- Mengizinkan pemulihan basis data Amazon RDS, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan pemilik tag mengakses sepenuhnya sumber daya Amazon RDS yang telah mereka tandai (Lihat kebijakan ini.)

Contoh kebijakan: Amazon S3

- Mengizinkan pengguna Amazon Cognito mengakses objek dalam bucket Amazon S3 milik mereka (Lihat kebijakan ini.)

- Memungkinkan pengguna dengan kredensi sementara untuk mengakses direktori home mereka sendiri di Amazon S3, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Mengizinkan akses S3 penuh, tetapi secara tegas menolak akses ke bucket Produksi jika administrator belum masuk menggunakan MFA dalam tiga puluh menit terakhir (Lihat kebijakan ini.)
- Mengizinkan pengguna IAM untuk mengakses direktori home milik mereka di Amazon S3, secara terprogram dan di konsol (Lihat kebijakan ini.)
- Memungkinkan pengguna mengelola satu bucket Amazon S3 dan menolak setiap AWS tindakan dan sumber daya lainnya (Lihat kebijakan ini.)
- Mengizinkan akses `Read` dan `Write` ke bucket Amazon S3 tertentu (Lihat kebijakan ini.)
- Mengizinkan akses `Read` dan `Write` ke bucket Amazon S3 tertentu, secara terprogram dan di konsol (Lihat kebijakan ini.)

AWS: Mengizinkan akses berdasarkan tanggal dan waktu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses ke tindakan berdasarkan tanggal dan waktu. Kebijakan ini membatasi akses ke tindakan yang terjadi antara 1 April 2020 dan 30 Juni 2020 (UTC), inklusif. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk mempelajari tentang menggunakan beberapa ketentuan dalam blok `Condition` kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
```

```
            }
        }
    ]
}
```

> **Note**
>
> Anda tidak dapat menggunakan variabel kebijakan dengan operator kondisi Tanggal. Untuk mempelajari lebih lanjut lihat Elemen kondisi

AWS: Memungkinkan mengaktifkan dan menonaktifkan Wilayah AWS

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan administrator mengaktifkan dan menonaktifkan Wilayah Asia Pasifik (Hong Kong) (ap-east-1). Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Pengaturan ini muncul di halaman Pengaturan akun di Konsol Manajemen AWS. Halaman ini mencakup informasi tingkat akun yang sensitif yang harus dilihat dan dikelola hanya oleh administrator akun. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Important**
>
> Anda tidak dapat mengaktifkan atau menonaktifkan wilayah yang diaktifkan secara default. Anda hanya dapat memasukkan wilayah yang dinonaktifkan secara default. Untuk informasi selengkapnya, lihat Mengelola Wilayah AWS di Referensi Umum AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
```

```
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM yang diautentikasi menggunakan otentikasi multi-faktor (MFA) untuk mengelola kredensialnya sendiri di halaman Kredensi Keamanan. Halaman Konsol Manajemen AWS ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, perangkat MFA, sertifikat X.509, dan kunci SSH serta kredensial Git milik mereka. Contoh kebijakan ini mencakup izin yang diperlukan untuk melihat dan mengubah semua informasi di halaman tersebut. Ini juga mengharuskan pengguna untuk mengatur dan mengotentikasi menggunakan MFA sebelum melakukan operasi lain di. AWS Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

> **Note**
>
> - Kebijakan contoh ini tidak mengizinkan pengguna untuk mengatur ulang kata sandi saat masuk Konsol Manajemen AWS untuk pertama kalinya. Kami menyarankan Anda

untuk tidak memberikan izin kepada pengguna baru sampai mereka masuk. Untuk informasi selengkapnya, lihat Bagaimana cara membuat pengguna IAM dengan aman?. Ini juga mencegah pengguna dengan kata sandi yang kedaluwarsa mengatur ulang kata sandi mereka saat masuk. Anda dapat mengizinkan ini dengan menambahkan `iam:ChangePassword` dan `iam:GetAccountPasswordPolicy` ke pernyataan `DenyAllExceptListedIfNoMFA`. Namun, kami tidak merekomendasikan ini karena mengizinkan pengguna untuk mengubah kata sandi mereka tanpa MFA dapat menjadi risiko keamanan.

- Jika Anda bermaksud menggunakan kebijakan ini untuk akses terprogram, Anda harus menelepon `GetSessionToken`untuk mengautentikasi dengan MFA. Untuk informasi selengkapnya, lihat Akses API aman dengan MFA.

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `ListVirtualMFADevices` – Melihat detail tentang perangkat MFA virtual yang diaktifkan untuk pengguna.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- Pernyataan `AllowManageOwnSigningCertificates` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus sertifikat tanda tangan milik mereka.
- Pernyataan `AllowManageOwnSSHPublicKeys` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH milik mereka untuk CodeCommit.

- Pernyataan `AllowManageOwnGitCredentials` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus kredensial Git milik mereka untuk CodeCommit.
- `AllowManageOwnVirtualMFADevice`Pernyataan tersebut memungkinkan pengguna untuk membuat perangkat MFA virtual mereka sendiri. ARN sumber daya dalam pernyataan ini memungkinkan pengguna untuk membuat perangkat MFA dengan nama apa pun, tetapi pernyataan lain dalam kebijakan hanya mengizinkan pengguna untuk melampirkan perangkat ke pengguna yang saat ini masuk.
- Pernyataan `AllowManageOwnUserMFA` mengizinkan pengguna melihat atau mengelola perangkat MFA virtual, U2F, atau perangkat keras untuk pengguna milik mereka. ARN sumber daya dalam pernyataan ini mengizinkan akses hanya ke pengguna IAM milik pengguna. Pengguna tidak dapat melihat atau mengelola perangkat MFA untuk pengguna lain.
- `DenyAllExceptListedIfNoMFA`Pernyataan tersebut menolak akses ke setiap tindakan di semua AWS layanan, kecuali beberapa tindakan yang terdaftar, tetapi hanya jika pengguna tidak masuk dengan MFA. Pernyataan ini menggunakan kombinasi `"Deny"` dan `"NotAction"` untuk secara tegas menolak akses ke setiap tindakan yang tidak tercantum. Item yang tercantum tidak ditolak atau diizinkan oleh pernyataan ini. Namun, tindakan tersebut diizinkan oleh pernyataan lain dalam kebijakan tersebut. Untuk informasi lebih lanjut tentang logika untuk pernyataan ini, lihat NotAction dengan Menolak. Jika pengguna masuk di dengan MFA, maka pengujian `Condition` gagal dan pernyataan ini tidak menolak tindakan apa pun. Dalam kasus ini, kebijakan atau pernyataan lain untuk pengguna menentukan izin pengguna.

  Pernyataan ini memastikan bahwa ketika pengguna belum masuk dengan MFA yang dapat mereka lakukan hanya tindakan yang tercantum. Sebagai tambahan, mereka dapat melakukan tindakan yang tercantum hanya jika pernyataan atau kebijakan lain mengizinkan akses ke tindakan tersebut. Ini tidak mengizinkan pengguna untuk membuat kata sandi saat masuk, karena tindakan `iam:ChangePassword` seharusnya tidak boleh diizinkan tanpa otorisasi MFA.

  Versi `...IfExists` dari operator `Bool` memastikan bahwa jika kunci `aw:MultiFactorAuthPresent` hilang, ketentuan menghasilkan sah. Ini berarti bahwa pengguna yang mengakses kredensial jangka panjang sebuah API, seperti kunci akses, tidak diberi akses ke operasi API non-IAM.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo` dan pernyataan `DenyAllExceptListedIfNoMFA`. Ini juga tidak mengizinkan pengguna

untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `AllowManageOwnPasswords` pernyataan. Untuk juga mengizinkan pengguna mengubah kata sandi mereka dari halaman pengguna milik mereka tanpa masuk menggunakan MFA, tambahkan tindakan `iam:UpdateLoginProfile` ke pernyataan `DenyAllExceptListedIfNoMFA`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
```

```
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

AWS: Mengizinkan akses tertentu dalam tanggal tertentu menggunakan MFA

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menggunakan beberapa kondisi, yang dievaluasi menggunakan logika. `AND` Ini memungkinkan akses penuh ke layanan bernama `SERVICE-NAME-1`, dan akses ke tindakan `ACTION-NAME-A` dan `ACTION-NAME-B` dalam layanan bernama `SERVICE-NAME-2`. Tindakan ini hanya diizinkan ketika pengguna diautentikasi menggunakan autentikasi multifaktor (MFA). Akses dibatasi untuk tindakan yang terjadi antara 1 Juli 2017 dan 31 Desember 2017 (UTC), termasuk. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS

CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk mempelajari tentang menggunakan beberapa ketentuan dalam blok `Condition` kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola semua kredensialnya sendiri di halaman kredensial Keamanan. Konsol Manajemen AWS Halaman ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, sertifikat X.509, kunci SSH, serta kredensial Git milik mereka. Contoh kebijakan ini mencakup izin yang diperlukan untuk melihat dan mengubah semua informasi di halaman tersebut kecuali perangkat MFA pengguna. Untuk mengizinkan pengguna mengelola kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

Untuk mempelajari cara pengguna mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `GetAccountSummary` – Melihat ID akun dan akun ID pengguna resmi.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- Pernyataan `AllowManageOwnSigningCertificates` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus sertifikat tanda tangan milik mereka.
- Pernyataan `AllowManageOwnSSHPublicKeys` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH milik mereka untuk CodeCommit.
- Pernyataan `AllowManageOwnGitCredentials` memungkinkan pengguna membuat, memperbarui, dan menghapus kredensial milik mereka untuk CodeCommit.

Kebijakan ini tidak mengizinkan pengguna melihat atau mengelola perangkat MFA milik mereka. Mereka juga tidak dapat melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile`, and `iam:UpdateLoginProfile` ke pernyataan `AllowManageOwnPasswords`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
```

```
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM yang diautentikasi melalui otentikasi multi-faktor (MFA) untuk mengelola perangkat MFA mereka sendiri di halaman kredensi Keamanan. Konsol Manajemen AWS Halaman ini menampilkan informasi akun dan pengguna, tetapi pengguna hanya dapat melihat dan mengedit perangkat MFA mereka sendiri. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

> **Note**
>
> Jika pengguna IAM dengan kebijakan ini tidak diautentikasi oleh MFA, kebijakan ini menolak akses ke semua AWS tindakan kecuali yang diperlukan untuk mengautentikasi menggunakan MFA. Untuk menggunakan AWS API AWS CLI dan, pengguna IAM harus terlebih dahulu mengambil token MFA mereka menggunakan AWS STS GetSessionTokenoperasi dan kemudian menggunakan token itu untuk mengautentikasi operasi yang diinginkan. Kebijakan lain, seperti kebijakan berbasis sumber daya atau kebijakan berbasis identitas lainnya dapat memungkinkan tindakan di layanan lain. Kebijakan ini akan menolak akses tersebut jika pengguna IAM tidak diautentikasi oleh MFA.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat detail tentang perangkat MFA virtual yang diaktifkan untuk pengguna. Izin ini harus berada dalam pernyataannya sendiri karena hal itu tidak mendukung ARN sumber daya tertentu. Sebagai gantinya Anda harus menentukan `"Resource" : "*"`.
- `AllowManageOwnVirtualMFADevice`Pernyataan tersebut memungkinkan pengguna untuk membuat perangkat MFA virtual mereka sendiri. ARN sumber daya dalam pernyataan ini memungkinkan pengguna untuk membuat perangkat MFA dengan nama apa pun, tetapi pernyataan lain dalam kebijakan hanya mengizinkan pengguna untuk melampirkan perangkat ke pengguna yang saat ini masuk.
- Pernyataan `AllowManageOwnUserMFA` mengizinkan pengguna untuk melihat atau mengelola perangkat MFA virtual, U2F, atau perangkat keras milik mereka. ARN sumber daya dalam pernyataan ini mengizinkan akses hanya ke pengguna IAM milik pengguna. Pengguna tidak dapat melihat atau mengelola perangkat MFA untuk pengguna lain.
- `DenyAllExceptListedIfNoMFA`Pernyataan tersebut menolak akses ke setiap tindakan di semua AWS layanan, kecuali beberapa tindakan yang terdaftar, tetapi hanya jika pengguna tidak masuk dengan MFA. Pernyataan ini menggunakan kombinasi `"Deny"` dan `"NotAction"` untuk secara tegas menolak akses ke setiap tindakan yang tidak tercantum. Item yang tercantum tidak ditolak atau diizinkan oleh pernyataan ini. Namun, tindakan tersebut diizinkan oleh pernyataan lain dalam kebijakan tersebut. Untuk informasi lebih lanjut tentang logika untuk pernyataan ini, lihat NotAction dengan Menolak. Jika pengguna masuk di dengan MFA, maka pengujian `Condition`

gagal dan pernyataan ini tidak menolak tindakan apa pun. Dalam kasus ini, kebijakan atau pernyataan lain untuk pengguna menentukan izin pengguna.

Pernyataan ini memastikan bahwa ketika pengguna belum masuk di dengan MFA, mereka hanya dapat melakukan tindakan yang tercantum. Sebagai tambahan, mereka dapat melakukan tindakan yang tercantum hanya jika pernyataan atau kebijakan lain mengizinkan akses ke tindakan tersebut.

Versi `...IfExists` dari operator `Bool` memastikan bahwa jika kunci `aws:MultiFactorAuthPresent` hilang, ketentuan menghasilkan sah. Ini berarti bahwa pengguna yang mengakses kredensial jangka panjang operasi API, seperti kunci akses, ditolak aksesnya ke operasi API non-IAM.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengizinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo` dan pernyataan `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> Jangan tambahkan izin untuk menghapus perangkat MFA tanpa autentikasi MFA. Pengguna dengan kebijakan ini mungkin mencoba untuk menetapkan sendiri perangkat MFA virtual dan menerima kesalahan bahwa mereka tidak berwenang untuk melakukan. `iam:DeleteVirtualMFADevice` Jika ini terjadi, jangan tambahkan izin tersebut ke pernyataan `DenyAllExceptListedIfNoMFA`. Pengguna yang tidak diautentikasi menggunakan MFA seharusnya tidak pernah diizinkan untuk menghapus perangkat MFA mereka. Pengguna mungkin melihat kesalahan ini jika sebelumnya mereka mulai memberikan perangkat MFA virtual ke penggunanya dan membatalkan proses tersebut. Untuk mengatasi masalah ini, Anda atau administrator lain harus menghapus perangkat MFA virtual pengguna yang ada menggunakan API AWS CLI atau AWS . Lihat informasi yang lebih lengkap di Saya tidak berwenang untuk melakukan: iam: DeleteVirtualMFADevice.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
```

```
    ]
}
```

AWS: Memungkinkan pengguna IAM untuk mengubah kata sandi konsol mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengubah Konsol Manajemen AWS kata sandi mereka sendiri di halaman kredensial Keamanan. Konsol Manajemen AWS Halaman ini menampilkan informasi akun dan pengguna, tetapi pengguna hanya dapat mengakses kata sandi mereka sendiri. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan. Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna dapat mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `ViewAccountPasswordRequirements` mengizinkan pengguna untuk melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
- Pernyataan `ChangeOwnPassword` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.

Kebijakan ini tidak mengizinkan pengguna untuk melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `ViewAccountPasswordRequirements`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `ChangeOwnPassword` pernyataan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Memungkinkan pengguna IAM untuk mengelola kata sandi, kunci akses, dan kunci publik SSH mereka sendiri di halaman kredensi Keamanan

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola kata sandi, kunci akses, dan sertifikat X.509 mereka sendiri di halaman kredensial Keamanan. Halaman Konsol Manajemen AWS ini menampilkan informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengubah kata sandi, access key, perangkat MFA, sertifikat X.509, kunci SSH, dan kredensial Git milik mereka. Kebijakan contoh ini mencakup izin yang diperlukan untuk hanya melihat dan mengubah kata sandi, access key, dan sertifikat X.509 miliknya. Untuk mengizinkan pengguna mengelola semua kredensial milik mereka menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan. Untuk mengizinkan pengguna mengelola kredensial milik mereka tanpa menggunakan MFA, lihat AWS: Memungkinkan pengguna IAM untuk mengelola kredensialnya sendiri di halaman kredensial Keamanan.

Untuk mempelajari cara pengguna mengakses halaman Kredensial keamanan, lihat. Cara pengguna IAM mengubah kata sandi mereka sendiri (konsol)

Apa yang dilakukan kebijakan ini?

- Pernyataan `AllowViewAccountInfo` mengizinkan pengguna untuk melihat informasi tingkat akun. Izin ini harus ada di dalam pernyataan milik mereka karena hal itu tidak mendukung atau tidak perlu untuk menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`. Pernyataan ini mencakup tindakan berikut yang mengizinkan pengguna melihat informasi spesifik:
  - `GetAccountPasswordPolicy` – Melihat persyaratan kata sandi akun saat mengubah kata sandi pengguna IAM milik mereka.
  - `GetAccountSummary` – Melihat ID akun dan akun ID pengguna resmi.
- Pernyataan `AllowManageOwnPasswords` mengizinkan pengguna untuk mengubah kata sandi milik mereka. Pernyataan ini juga mencakup `GetUser` tindakan, yang diperlukan untuk melihat sebagian besar informasi di halaman Kredensi Keamanan Saya.
- Pernyataan `AllowManageOwnAccessKeys` mengizinkan pengguna untuk membuat, memperbarui, dan menghapus access key milik mereka. Pengguna juga dapat mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan.
- Pernyataan `AllowManageOwnSSHPublicKeys` mengizinkan pengguna untuk mengunggah, memperbarui, dan menghapus kunci publik SSH milik mereka untuk CodeCommit.

Kebijakan ini tidak mengizinkan pengguna melihat atau mengelola perangkat MFA milik mereka. Mereka juga tidak dapat melihat halaman Pengguna di konsol IAM atau menggunakan halaman tersebut untuk mengakses informasi pengguna milik mereka. Untuk mengijinkan ini, tambahkan tindakan `iam:ListUsers` ke pernyataan `AllowViewAccountInfo`. Ini juga tidak mengizinkan pengguna untuk mengubah kata sandi mereka di halaman pengguna milik mereka. Untuk memungkinkan ini, tambahkan `iam:GetLoginProfile` dan `iam:UpdateLoginProfile` tindakan ke `AllowManageOwnPasswords` pernyataan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: Menolak akses AWS berdasarkan Wilayah yang diminta

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang menolak akses ke tindakan apa pun di luar Wilayah yang ditentukan menggunakan kunci `aws:RequestedRegion` kondisi, kecuali untuk tindakan dalam layanan yang ditentukan menggunakan. `NotAction` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini menggunakan elemen `NotAction` dengan efek `Deny`, yang secara tegas menghalangi akses ke semua tindakan yang tidak tercantum dalam pernyataan. Tindakan dalam CloudFront, IAM, Rute 53, dan Dukungan layanan tidak boleh ditolak karena ini adalah layanan AWS global populer dengan titik akhir tunggal yang secara fisik terletak di `us-east-1` Wilayah. Karena semua permintaan untuk layanan ini diajukan ke Wilayah `us-east-1`, permintaan tidak akan ditolak tanpa elemen `NotAction`. Edit elemen ini untuk menyertakan tindakan untuk layanan AWS global lainnya yang Anda gunakan, sepertibudgets,`globalaccelerator`,`importexport`,`organizations`, atau`waf`. Beberapa layanan global lainnya, seperti Amazon Q Developer dalam aplikasi obrolan dan AWS Device Farm, adalah layanan global dengan titik akhir yang secara fisik berlokasi di `us-west-2` wilayah tersebut. Untuk mempelajari tentang semua layanan yang memiliki titik akhir global tunggal, lihatWilayah dan Titik Akhir AWS di Referensi Umum AWS. Untuk informasi selengkapnya tentang penggunaan elemen `NotAction` dengan efek `Deny`, lihat Elemen kebijakan IAM JSON: NotAction.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
```

```
                "cloudfront:*",
                "iam:*",
                "organizations:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS: Menolak akses AWS berdasarkan IP sumber

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menolak akses ke semua AWS tindakan di akun saat permintaan berasal dari prinsipal di luar rentang IP yang ditentukan. Kebijakan ini berguna saat alamat IP untuk perusahaan Anda berada dalam cakupan tertentu. Dalam contoh ini, permintaan akan ditolak kecuali berasal dari kisaran CIDR 192.0.2.0/24 atau 203.0.113.0/24. Kebijakan ini tidak menolak permintaan yang dibuat oleh AWS layanan yang digunakan Teruskan sesi akses karena alamat IP pemohon asli dipertahankan.

Hati-hati saat menggunakan kondisi negatif dalam pernyataan kebijakan yang sama seperti `"Effect": "Deny"`. Saat Anda melakukannya, tindakan yang ditentukan dalam pernyataan kebijakan secara eksplisit ditolak di semua kondisi kecuali untuk yang ditentukan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

Ketika kebijakan lain mengizinkan tindakan, prinsipal dapat membuat permintaan dari rentang alamat IP. AWS Layanan juga dapat membuat permintaan menggunakan kredensi kepala sekolah. Saat prinsipal mengajukan permintaan dari luar rentang IP, permintaan tersebut ditolak.

Untuk informasi selengkapnya tentang penggunaan kunci `aws:SourceIp` kondisi, termasuk informasi tentang kapan `aws:SourceIp` mungkin tidak berfungsi dalam kebijakan Anda, lihatAWS kunci konteks kondisi global.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang menolak akses ke semua sumber daya AWS yang bukan milik akun Anda, kecuali sumber daya yang AWS Data Exchange diperlukan untuk pengoperasian normal. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam organisasi atau unit organisasi, sambil menghitung sumber daya yang AWS Data Exchange dimiliki dengan menggunakan kunci kondisi `aws:ResourceOrgPaths` dan`aws:ResourceOrgID`.

Jika Anda menggunakan AWS Data Exchange di lingkungan Anda, layanan akan membuat dan berinteraksi dengan sumber daya seperti bucket Amazon S3 yang dimiliki oleh akun layanan. Misalnya, AWS Data Exchange mengirimkan permintaan ke bucket Amazon S3 yang dimiliki oleh AWS Data Exchange layanan atas nama prinsipal IAM (pengguna atau peran) yang memanggil. AWS Data Exchange APIs Dalam hal ini, menggunakan`aws:ResourceAccount`,`aws:ResourceOrgPaths`, atau `aws:ResourceOrgID` dalam kebijakan, tanpa memperhitungkan sumber daya yang AWS Data Exchange dimiliki, menolak akses ke ember yang dimiliki oleh akun layanan.

- Pernyataan tersebut`DenyAllAwsResourcesOutsideAccountExceptS3`,, menggunakan `NotAction` elemen dengan efek Deny yang secara eksplisit menolak akses ke setiap tindakan yang tidak tercantum dalam pernyataan yang juga bukan milik akun yang terdaftar. `NotAction`Elemen menunjukkan pengecualian untuk pernyataan ini. Tindakan ini merupakan pengecualian untuk pernyataan ini karena jika tindakan dilakukan pada sumber daya yang dibuat oleh AWS Data Exchange, kebijakan menyangkalnya.
- Pernyataan itu`DenyAllS3ResourcesOutsideAccountExceptDataExchange`,, menggunakan kombinasi `ResourceAccount` dan `CalledVia` kondisi untuk menolak akses ke tiga tindakan Amazon S3 yang dikecualikan dalam pernyataan sebelumnya. Pernyataan tersebut menyangkal tindakan jika sumber daya tidak termasuk dalam akun yang terdaftar dan jika layanan panggilan tidak AWS Data Exchange. Pernyataan tersebut tidak menyangkal tindakan jika sumber daya milik akun yang terdaftar atau kepala layanan yang terdaftar`dataexchange.amazonaws.com`,, melakukan operasi.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Ini menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

Contoh berikut menunjukkan cara mengonfigurasi kebijakan untuk mengizinkan akses ke bucket Amazon S3 yang diperlukan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
```

```
    }
  ]
}
```

AWS Data Pipeline: Menolak akses ke DataPipeline saluran pipa yang tidak dibuat pengguna

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang menolak akses ke pipeline yang tidak dibuat pengguna. Jika nilai kolom `PipelineCreator` cocok dengan nama pengguna IAM, maka tindakan yang ditentukan tidak ditolak. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
```

```
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator":
 "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB: Memungkinkan akses ke tabel tertentu

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke tabel `MyTable` DynamoDB. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Important**
>
> Kebijakan ini memungkinkan semua tindakan yang dapat dilakukan pada tabel DynamoDB. Untuk meninjau tindakan ini, lihat Izin API DynamoDB: Referensi Tindakan, Sumber Daya, dan Ketentuan di Panduan Pengembang Amazon DynamoDB. Anda dapat memberikan izin yang sama dengan mencantumkan setiap tindakan individu. Namun, jika Anda menggunakan wildcard (*) dalam `Action` elemen, misalnya, Anda tidak perlu memperbarui kebijakan jika DynamoDB menambahkan tindakan List baru. `"dynamodb:List*"`

Kebijakan ini mengizinkan tindakan hanya pada tabel DynamoDB yang ada dengan nama yang ditentukan. Untuk memungkinkan pengguna `Read` mengakses semua yang ada di DynamoDB, Anda juga dapat melampirkan AmazonDynamoDBReadOnlyAccess AWS kebijakan terkelola.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB: Memungkinkan akses ke atribut tertentu

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan akses ke atribut DynamoDB tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Persyaratan `dynamodb:Select` mencegah tindakan API dari mengembalikan atribut apapun yang tidak diperbolehkan, seperti dari proyeksi indeks. Untuk mempelajari selengkapnya tentang kunci kondisi DynamoDB, lihat Menentukan Kondisi: Menggunakan Kunci Kondisi di Panduan Pengembang Amazon DynamoDB. Untuk mempelajari tentang menggunakan beberapa ketentuan atau beberapa kunci kondisi dalam blok `Condition` dari kebijakan IAM, lihat Beberapa nilai dalam suatu ketentuan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select":
 "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB: Mengizinkan akses tingkat item ke DynamoDB berdasarkan ID Amazon Cognito

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan akses tingkat item ke tabel `MyTable` DynamoDB berdasarkan ID pengguna kumpulan identitas Amazon Cognito. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk menggunakan kebijakan ini, Anda harus menyusun tabel DynamoDB sehingga ID pengguna kumpulan identitas Amazon Cognito adalah kunci partisi. Untuk informasi selengkapnya, lihat Membuat Tabel di Panduan Pengembang Amazon DynamoDB.

Untuk mempelajari selengkapnya tentang kunci kondisi DynamoDB, lihat Menentukan Kondisi: Menggunakan Kunci Kondisi di Panduan Pengembang Amazon DynamoDB.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-
identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
```

```
}
```

Amazon EC2: Lampirkan atau lepaskan volume Amazon EBS ke instans EC2 berdasarkan tag

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemilik volume EBS melampirkan atau melepaskan volume EBS mereka yang ditentukan menggunakan tag `VolumeUser` ke instans EC2 yang ditandai sebagai instance pengembangan (). `Department=Development` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Untuk informasi selengkapnya tentang membuat kebijakan IAM untuk mengontrol akses ke sumber daya Amazon EC2, lihat Mengontrol Akses ke Sumber Daya Amazon EC2 di Panduan Pengguna Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
```

```
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2: Memungkinkan peluncuran instans EC2 di subnet tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan daftar informasi untuk semua objek EC2 dan meluncurkan instans EC2 di subnet tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
```

```
            ]
        }
    ]
}
```

Amazon EC2: Memungkinkan mengelola grup keamanan EC2 dengan pasangan nilai kunci tag tertentu secara terprogram dan di konsol

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memberikan izin kepada pengguna untuk mengambil tindakan tertentu untuk grup keamanan yang memiliki tag yang sama. Kebijakan ini memberikan izin untuk melihat grup keamanan di konsol Amazon EC2, menambah dan menghapus aturan masuk dan keluar, serta mencantumkan serta mengubah deskripsi aturan untuk grup keamanan yang ada dengan tag. `Department=Test` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
         "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
         "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
```

```
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:security-group/*"
      ],
      "Condition": {
         "StringEquals": {
            "aws:ResourceTag/Department": "Test"
         }
      }
     },
     {
      "Effect": "Allow",
      "Action": [
         "ec2:ModifySecurityGroupRules"
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1:111122223333:security-group-rule/*"
      ]
     }
   ]
}
```

Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 yang telah ditandai pengguna, secara terprogram, dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk memulai atau menghentikan instans EC2, tetapi hanya jika tag instans `Owner` memiliki nilai nama pengguna tersebut. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
```

```
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2: Mulai atau hentikan instance berdasarkan tag

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instance dengan pasangan nilai kunci tag`Project = DataAnalytics`, tetapi hanya berdasarkan prinsip dengan pasangan nilai kunci tag. `Department = Data` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Ketentuan dalam pengembalian kebijakan benar jika kedua bagian dari ketentuan tersebut benar. Instans harus memiliki tanda `Project=DataAnalytics`. Sebagai tambahan, prinsipal IAM (pengguna atau peran) yang membuat permintaan harus memiliki tag `Department=Data`.

> **Note**
>
> Sebagai praktik terbaik, lampirkan kebijakan dengan kunci ketentuan `aws:PrincipalTag` ke grup IAM, untuk kasus di mana beberapa pengguna mungkin memiliki tag tertentu dan beberapa mungkin tidak.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2: Mulai atau hentikan instans berdasarkan pada pencocokan prinsipal dan tanda sumber daya

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan prinsipal memulai atau menghentikan instans Amazon EC2 saat tag sumber daya instans dan tag prinsipal memiliki nilai yang sama untuk kunci tag. `CostCenter` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

> **Note**
>
> Sebagai praktik terbaik, lampirkan kebijakan dengan kunci ketentuan `aws:PrincipalTag` ke grup IAM, untuk kasus di mana beberapa pengguna mungkin memiliki tag tertentu dan beberapa mungkin tidak.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2: Memungkinkan akses EC2 penuh dalam Wilayah tertentu, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses EC2 penuh dalam Wilayah tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan. Untuk daftar kode Wilayah, lihat Wilayah yang Tersedia dalam Panduan Pengguna Amazon EC2.

Atau, Anda dapat menggunakan kunci kondisi global `aws:RequestedRegion`, yang didukung oleh semua tindakan Amazon EC2 API. Untuk informasi selengkapnya, lihat Contoh: Membatasi akses ke Wilayah tertentu di Panduan Pengguna Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
```

```
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2: Memungkinkan memulai atau menghentikan instans EC2 dan memodifikasi grup keamanan, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan memulai atau menghentikan instans EC2 tertentu dan memodifikasi grup keamanan tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
```

```
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2: Memerlukan MFA (GetSessionToken) untuk operasi EC2 tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke semua operasi AWS API di Amazon EC2. Namun, itu secara tegas menolak akses ke `StopInstances` dan `TerminateInstances` Operasi API jika pengguna tidak diautentikasi menggunakan Multi-Factor Authentication (MFA). Untuk melakukan ini secara terprogram, pengguna harus menyertakan nilai opsional `SerialNumber` dan `TokenCode` saat memanggil operasi `GetSessionToken`. Operasi ini mengembalikan kredensial sementara yang diautentikasi menggunakan MFA. Untuk mempelajari lebih lanjut tentang GetSessionToken, lihatMeminta kredensi untuk pengguna di lingkungan yang tidak tepercaya.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `AllowAllActionsForEC2` mengizinkan semua tindakan Amazon EC2
- Pernyataan `DenyStopAndTerminateWhenMFAIsNotPresent` menolak tindakan `StopInstances` dan `TerminateInstances` saat konteks MFA hilang. Ini berarti tindakan tersebut ditolak saat konteks multi-factor authentication hilang (artinya MFA tidak digunakan). Penolakan mennimpa izin.

> **Note**
>
> Pemeriksaaan ketentuan untuk `MultiFactorAuthPresent` dalam pernyataan `Deny` tidak boleh berupa `{"Bool":{"aws:MultiFactorAuthPresent":false}}` karena kunci tersebut tidak ada dan tidak dapat dievaluasi saat MFA tidak digunakan. Sebagai gantinya, gunakan periksa `BoolIfExists` untuk melihat apakah kuncinya ada sebelum memeriksa nilai. Lihat informasi yang lebih lengkap di ... IfExists operator kondisi.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2: Membatasi penghentian instans EC2 ke rentang alamat IP

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang membatasi instans EC2 dengan mengizinkan tindakan, tetapi secara eksplisit menolak akses ketika permintaan berasal dari luar rentang IP yang ditentukan. Kebijakan ini berguna saat alamat IP untuk perusahaan Anda berada dalam cakupan tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Jika kebijakan ini digunakan bersama dengan kebijakan lain yang mengizinkan `ec2:TerminateInstances` tindakan (seperti kebijakan yang EC2 FullAccess AWS dikelola Amazon), akses ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit lebih diutamakan

daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.

> **Important**
>
> Kunci `aws:SourceIp` kondisi menolak akses ke Layanan AWS, seperti AWS CloudFormation, yang membuat panggilan atas nama Anda. Untuk informasi selengkapnya tentang penggunaan kunci kondisi `aws:SourceIp`, lihat AWS kunci konteks kondisi global.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM: Akses API simulator kebijakan

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan API simulator kebijakan untuk kebijakan yang dilampirkan ke pengguna, grup, atau

peran saat ini. Akun AWS Kebijakan ini juga mengizinkan akses untuk menyimulasikan kebijakan yang kurang sensitif untuk diteruskan ke API sebagai strings. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Untuk mengizinkan pengguna mengakses konsol simulator kebijakan guna mensimulasikan kebijakan yang dilampirkan ke pengguna, grup, atau peran saat ini Akun AWS, lihatIAM: Akses konsol simulator kebijakan.

IAM: Akses konsol simulator kebijakan

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan konsol simulator kebijakan untuk kebijakan yang dilampirkan ke pengguna, grup, atau peran saat ini. Akun AWS

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "access-analyzer:ValidatePolicy",
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListPolicies",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM: Asumsikan peran yang memiliki tag tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk mengambil peran dengan pasangan nilai kunci tag. `Project = ExampleCorpABC` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Jika ada peran dengan tag ini pada akun yang sama dengan pengguna, maka pengguna dapat mengambil peran tersebut. Jika ada peran dengan tanda ini pada akun selain milik pengguna, peran tersebut memerlukan izin tambahan. Kebijakan kepercayaan peran lintas-akun juga harus mengizinkan pengguna atau semua anggota dari akun pengguna untuk mengambil peran itu. Untuk informasi tentang penggunaan peran untuk akses lintas-akun, lihat Akses untuk pengguna IAM di yang lain Akun AWS yang Anda miliki.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: Mengizinkan dan menolak akses ke beberapa layanan secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses penuh ke beberapa layanan dan akses pengelolaan mandiri terbatas di IAM. Ia juga menolak akses ke bucket `logs` Amazon S3 atau contoh `i-1234567890abcdef0` Amazon EC2. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

> **Warning**
>
> Kebijakan ini mengizinkan akses penuh ke setiap tindakan dan sumber daya dalam beberapa layanan. Kebijakan ini harus diterapkan hanya bagi administrator yang tepercaya.

Anda dapat menggunakan kebijakan ini sebagai batasan izin untuk menentukan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada pengguna IAM. Untuk informasi selengkapnya, lihat Mendelegasikan tanggung jawab kepada orang lain menggunakan batas izin. Ketika kebijakan itu digunakan sebagai batasan izin bagi pengguna, pernyataan itu menentukan batasan berikut:

- Pernyataan `AllowServices` mengizinkan akses penuh ke layanan AWS tertentu. Artinya, tindakan pengguna dalam layanan ini hanya dibatasi oleh kebijakan izin yang melekat pada pengguna.
- Pernyataan `AllowIAMConsoleForCredentials` mengizinkan akses untuk mencantumkan semua pengguna IAM. Akses ini diperlukan untuk menavigasi halaman Pengguna di Konsol Manajemen AWS. Ia juga mengizinkan untuk melihat persyaratan kata sandi untuk akun, yang diperlukan bagi pengguna untuk mengubah kata sandinya sendiri.
- Pernyataan `AllowManageOwnPasswordAndAccessKeys` ini mengizinkan pengguna mengelola kata sandi konsol dan kunci akses programnya sendiri. Ini penting karena jika kebijakan lain memberi akses IAM penuh ke pengguna, pengguna tersebut lalu dapat mengubah izinnya sendiri atau izin pengguna lain. Pernyataan ini mencegah hal tersebut terjadi.
- Pernyataan `DenyS3Logs` itu secara eksplisit menolak akses ke `logs` bucket. Kebijakan ini menerapkan batasan perusahaan kepada pengguna.
- Pernyataan `DenyEC2Production` secara eksplisit menolak akses ke instans `i-1234567890abcdef0`.

Kebijakan ini tidak mengizinkan akses ke layanan atau tindakan lain. Jika kebijakan digunakan sebagai batas izin pada pengguna, meskipun kebijakan lain yang dilampirkan pada pengguna mengizinkan tindakan tersebut, AWS menolak permintaan tersebut.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
```

```
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM: Tambahkan tag tertentu ke pengguna dengan tag tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penambahan kunci tag `Department` dengan nilai tag `MarketingDevelopment`, atau `QualityAssurance` ke pengguna IAM. Pengguna tersebut harus sudah menyertakan pasangan kunci tag — nilai. `JobFunction = Manager` Anda dapat menggunakan kebijakan ini untuk mewajibkan satu manajer hanya dimiliki oleh satu dari tiga departemen. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Pernyataan `ListTagsForAllUsers` memungkinkan melihat tanda untuk semua pengguna di akun Anda.

Kondisi pertama dalam `TagManagerWithSpecificDepartment` pernyataan menggunakan `StringEquals` operator kondisi. Kondisi akan kembali sebagai benar bila kedua bagian dari kondisi adalah benar. Pengguna yang akan ditandai harus sudah memiliki tanda `JobFunction=Manager`. Permintaan itu harus menyertakan kunci tanda `Department` dengan satu nilai tanda yang terdaftar.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi mengembalikan benar jika semua kunci tanda dalam permintaan sesuai dengan kunci dalam kebijakan. Artinya satu-satunya kunci tanda dalam permintaan tersebut harus `Department`. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Mengatur operator untuk kunci konteks multinilai.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM: Tambahkan tag tertentu dengan nilai tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan hanya menambahkan kunci tag `CostCenter` dan nilai tag `A-123` atau nilai tag `B-456` ke pengguna atau peran IAM mana pun. Anda dapat menggunakan kebijakan ini untuk membatasi penandaan ke kunci tanda tertentu dan serangkaian nilai tanda. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Pernyataan `ConsoleDisplay` memungkinkan melihat tanda untuk semua pengguna dan peran dalam akun Anda.

Kondisi pertama dalam `AddTag` pernyataan menggunakan `StringEquals` operator kondisi. Kondisi mengembalikan benar jika permintaan memasukkan kunci tanda `CostCenter` dengan satu nilai tanda yang terdaftar.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi mengembalikan benar jika semua kunci tanda dalam permintaan sesuai dengan kunci dalam kebijakan. Artinya satu-satunya kunci tanda dalam permintaan tersebut harus `CostCenter`. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Mengatur operator untuk kunci konteks multinilai.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM: Buat pengguna baru hanya dengan tag tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pembuatan pengguna IAM tetapi hanya dengan satu atau kedua kunci dan tag. `Department` `JobFunction` Kunci tanda `Department` harus memiliki nilai tanda `Development` atau `QualityAssurance`. Kunci tanda `JobFunction` harus memiliki nilia tanda `Employee`. Anda dapat menggunakan kebijakan ini untuk mewajibkan bahwa pengguna baru memiliki fungsi dan departemen tugas tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kondisi pertama dalam pernyataan itu menggunakan `StringEqualsIfExists` operator kondisi. Jika tanda dengan kunci `Department` atau `JobFunction` ada dalam permintaan, tanda tersebut harus memiliki nilai yang ditentukan. Jika tidak ada kunci yang muncul, maka kondisi ini akan dievaluasi sebagai benar. Satu-satunya cara bahwa kondisi itu dievaluasi sebagai salah adalah bila satu dari kunci kondisi yang ditentukan muncul dalam permintaan, tetapi memiliki nilai berbeda dari yang diizinkan. Untuk informasi selengkapnya tentang penggunaan `IfExists`, lihat ... IfExists operator kondisi.

Kondisi kedua menggunakan operator kondisi `ForAllValues:StringEquals`. Kondisi akan mengembalikan benar bila ada kecocokan di antara setiap kunci tanda khusus yang ditentukan dalam permintaan, dan setidaknya satu nilai dalam kebijakan. Artinya, semua tanda dalam permintaan harus ada dalam daftar ini. Namun, permintaan tersebut hanya dapat memasukkan satu tanda ke dalam daftar. Misalnya, Anda dapat membuat pengguna IAM hanya dengan `Department=QualityAssurance` tag. Namun, Anda tidak dapat membuat pengguna IAM dengan `JobFunction=Employee` tag dan `Project=core` tag. Untuk informasi selengkapnya tentang penggunaan `ForAllValues`, lihat Mengatur operator untuk kunci konteks multinilai.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
```

```
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: Menghasilkan dan mengambil laporan kredensi IAM

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna membuat dan mengunduh laporan yang mencantumkan semua pengguna IAM di dalamnya. Akun AWS Laporan ini termasuk status dari kredensial pengguna, termasuk kata sandi, access key, perangkat MFA, dan sertifikat tanda tangan. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

Untuk informasi selengkapnya tentang laporan kredensial, lihat Hasilkan laporan kredensi untuk Anda Akun AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
```

```
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
}
```

IAM: Memungkinkan mengelola keanggotaan grup secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan memperbarui keanggotaan grup yang dipanggil. `MarketingTeam` Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `ViewGroups` memungkinkan penggunanya mencantumkan semua pengguna dan grup di Konsol Manajemen AWS. Ia juga memungkinkan pengguna untuk melihat informasi dasar tentang pengguna di akun. Izin ini harus ada di dalam pernyataannya sendiri karena mereka tidak mendukung atau tidak perlu menentukan ARN sumber daya tertentu. Alih-alih, izin ini menentukan `"Resource" : "*"`.
- Pernyataan `ViewEditThisGroup` ini memungkinkan pengguna melihat informasi tentang `MarketingTeam` grup, dan menambahkan dan menghapus pengguna dari grup tersebut.

Kebijakan ini tidak mengizinkan pengguna untuk melihat atau mengedit izin dari pengguna atau `MarketingTeam` dari grup.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
```

```
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM: Kelola tag tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penambahan dan penghapusan tag IAM dengan kunci tag `Department` dari entitas IAM (pengguna dan peran). Kebijakan ini tidak membatasi nilai `Department` tag. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
```

```
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM: Lulus peran IAM ke layanan tertentu AWS

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan meneruskan peran layanan IAM apa pun ke layanan Amazon. CloudWatch Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Peran layanan adalah peran IAM yang menentukan AWS layanan sebagai prinsipal yang dapat mengambil peran tersebut. Ini memungkinkan layanan untuk mengambil peran dan mengakses sumber daya di layanan lain atas nama Anda. Untuk memungkinkan Amazon CloudWatch mengambil peran yang Anda lewati, Anda harus menentukan prinsip `cloudwatch.amazonaws.com` layanan sebagai prinsipal dalam kebijakan kepercayaan peran Anda. Prinsipiel layanan ditentukan oleh layanan tersebut. Untuk mempelajari prinsipal layanan dari layanan, lihat dokumentasi untuk layanan tersebut. Untuk beberapa layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang Ya dalam kolom Peran Terkait-Layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran yang terkait dengan layanan untuk layanan tersebut. Cari `amazonaws.com` untuk melihat prinsipal layanan.

Untuk mempelajari penerusan peran layanan ke layanan, lihat Memberikan izin pengguna untuk meneruskan peran ke AWS layanan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService":
 "cloudwatch.amazonaws.com"}
```

```
            }
        }
    ]
}
```

IAM: Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM untuk melakukan tindakan IAM apa pun yang dimulai dengan string atau. `Get` `List` Saat pengguna bekerja dengan konsol, konsol membuat permintaan ke IAM untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk menghasilkan laporan tentang sumber daya tersebut.

Tanda bintang berfungsi sebagai wildcard. Saat Anda menggunakan `iam:Get*` kebijakan, izin yang dihasilkan mencakup semua tindakan IAM yang dimulai`Get`, seperti `GetUser` dan. `GetRole` Wildcard berguna jika jenis entitas baru ditambahkan ke IAM di masa mendatang. Dalam hal itu, izin yang diberikan oleh kebijakan secara otomatis memungkinkan pengguna mendaftar dan mendapat rincian mengenai entitas baru tersebut.

Kebijakan ini tidak dapat digunakan untuk membuat laporan atau detail layanan yang terakhir diakses. Untuk kebijakan yang berbeda yang mengizinkan ini, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM.

JSON

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

IAM: Mengizinkan akses hanya-baca ke konsol IAM

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM melakukan tindakan IAM apa pun yang dimulai dengan string,, atau. `Get` `List` `Generate` Saat pengguna bekerja dengan konsol IAM, konsol membuat permintaan untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk menghasilkan laporan tentang sumber daya tersebut.

Tanda bintang berfungsi sebagai wildcard. Saat Anda menggunakan `iam:Get*` kebijakan, izin yang dihasilkan mencakup semua tindakan IAM yang dimulai`Get`, seperti `GetUser` dan. `GetRole` Menggunakan wildcard bermanfaat, terutama jika jenis entitas baru ditambahkan ke IAM di masa depan. Dalam hal itu, izin yang diberikan oleh kebijakan secara otomatis memungkinkan pengguna mendaftar dan mendapat rincian mengenai entitas baru tersebut.

Gunakan kebijakan ini untuk akses konsol yang menyertakan izin untuk menghasilkan laporan atau detail layanan yang terakhir diakses. Untuk kebijakan yang berbeda yang tidak memungkinkan pembuatan tindakan, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM tanpa melaporkan.

JSON

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

IAM: Memungkinkan pengguna IAM tertentu untuk mengelola grup secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM tertentu mengelola grup. `AllUsers` Kebijakan ini menetapkan izin untuk akses terprogram dan

konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Apa yang dikerjakan oleh kebijakan ini?

- Pernyataan `AllowAllUsersToListAllGroups` ini memungkinkan pendaftaran semua grup. Hal ini diperlukan untuk akses konsol. Izin ini harus berada dalam pernyataannya sendiri karena tidak mendukung ARN sumber daya. Alih-alih, izin ini menentukan `"Resource" : "*"`.
- Pernyataan `AllowAllUsersToViewAndManageThisGroup` ini mengizinkan semua tindakan grup yang dapat dilakukan pada tipe sumber daya grup. Ia tidak mengizinkan `ListGroupsForUser` tindakan, yang dapat dilakukan pada tipe sumber daya pengguna dan bukan tipe sumber daya grup. Untuk informasi selengkapnya tentang jenis sumber daya yang dapat Anda tentukan untuk tindakan IAM, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management.
- `LimitGroupManagementAccessToSpecificUsers`Pernyataan tersebut menolak pengguna dengan akses nama yang ditentukan untuk menulis dan tindakan grup manajemen izin. Ketika pengguna yang ditentukan dalam kebijakan mencoba untuk membuat perubahan pada grup, pernyataan ini tidak menolak permintaan tersebut. Permintaan tersebut akan diizinkan oleh `AllowAllUsersToViewAndManageThisGroup` pernyataan. Jika pengguna lain mencoba melakukan operasi ini, permintaannya ditolak. Anda dapat melihat tindakan IAM yang dijelaskan dengan Tulis atau pengelolaan izin tingkat akses sambil membuat kebijakan ini di konsol IAM. Untuk melakukan hal ini, beralihlah dari tab JSON ke tabeditor Visual. Untuk informasi selengkapnya tentang tingkat akses, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Identity and Access Management.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
```

```
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

IAM: Memungkinkan pengaturan persyaratan kata sandi akun secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna untuk melihat dan memperbarui persyaratan kata sandi akun mereka. Persyaratan kata sandi tersebut menentukan kompleksitas persyaratan dan periode rotasi wajib untuk kata sandi para anggota akun. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

Untuk mempelajari bagaimana mengatur persyaratan kata sandi akun untuk akun Anda, lihat Menetapkan kebijakan kata sandi akun untuk pengguna IAM.

JSON

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

IAM: Akses API simulator kebijakan berdasarkan jalur pengguna

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan API simulator kebijakan hanya untuk pengguna yang memiliki jalur. `Department/Development` Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
```

```
    ]
}
```

> **Note**
>
> Untuk membuat kebijakan yang memungkinkan penggunaan konsol simulator kebijakan untuk pengguna yang memiliki jalur `Department/Development`, lihat IAM: Akses konsol simulator kebijakan berdasarkan jalur pengguna.

IAM: Akses konsol simulator kebijakan berdasarkan jalur pengguna

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan konsol simulator kebijakan hanya untuk pengguna yang memiliki jalur tersebut. `Department/Development` Untuk menggunakan kebijakan ini, ganti kebijakan *`italicized placeholder text`* dalam contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
```

```
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM: Memungkinkan pengguna IAM untuk mengelola sendiri perangkat MFA

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengelola sendiri perangkat autentikasi multi-faktor (MFA) mereka. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

> **Note**
>
> Jika pengguna IAM dengan kebijakan ini tidak diautentikasi oleh MFA, kebijakan ini menolak akses ke semua AWS tindakan kecuali yang diperlukan untuk mengautentikasi menggunakan MFA. Jika Anda menambahkan izin ini untuk pengguna yang masuk AWS, mereka mungkin perlu keluar dan masuk kembali untuk melihat perubahan ini.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
```

```
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
```

```
            }
        }
    ]
}
```

IAM: Memungkinkan pengguna IAM memperbarui kredensialnya sendiri secara terprogram dan di konsol

Contoh ini menunjukkan cara Anda membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM memperbarui kunci akses mereka sendiri, menandatangani sertifikat, kredensi khusus layanan, dan kata sandi. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Untuk mempelajari bagaimana pengguna dapat mengubah kata sandinya sendiri di konsol, lihat the section called “Bagaimana pengguna IAM mengubah kata sandi mereka sendiri”.

IAM: Lihat informasi layanan yang terakhir diakses untuk kebijakan AWS Organizations

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan layanan melihat informasi yang terakhir diakses untuk kebijakan tertentu. AWS Organizations Kebijakan ini memungkinkan pengambilan data untuk kebijakan kendali layanan (SCP) dengan `p-policy123` ID. Orang yang membuat dan melihat laporan harus diautentikasi menggunakan kredensi akun AWS Organizations manajemen. Kebijakan ini memungkinkan pemohon untuk mengambil data untuk setiap AWS Organizations entitas dalam organisasi mereka. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Untuk informasi penting tentang informasi yang diakses terakhir, termasuk izin yang diperlukan, pemecahan masalah, dan Wilayah yang didukung, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
```

```
            }
        }
    ]
}
```

IAM: Membatasi kebijakan terkelola yang dapat diterapkan pada pengguna, grup, atau peran IAM

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang membatasi kebijakan yang dikelola dan AWS dikelola pelanggan yang dapat diterapkan pada pengguna, grup, atau peran IAM. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: Tolak akses ke sumber daya di luar akun Anda kecuali kebijakan IAM yang AWS dikelola

Menggunakan `aws:ResourceAccount` kebijakan berbasis identitas Anda dapat memengaruhi pengguna atau kemampuan peran untuk memanfaatkan beberapa layanan yang memerlukan interaksi dengan sumber daya di akun yang dimiliki oleh layanan.

Anda dapat membuat kebijakan dengan pengecualian untuk mengizinkan kebijakan IAM AWS terkelola. Akun yang dikelola layanan di luar Kebijakan IAM AWS Organizations Terkelola milik Anda sendiri. Ada empat tindakan IAM yang mencantumkan dan mengambil kebijakan AWS-managed. Gunakan tindakan ini dalam `NotAction`elemen pernyataan. `AllowAccessToResourcesInSpecificAccountsAndSpecificService1`dalam kebijakan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda: Mengizinkan fungsi Lambda mengakses tabel Amazon DynamoDB

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan akses baca dan tulis ke tabel Amazon DynamoDB tertentu. Kebijakan ini juga memungkinkan penulisan file log ke CloudWatch Log. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

Untuk menggunakan kebijakan ini, lampirkan kebijakan tersebut pada peran layanan Lambda. Peran layanan adalah peran yang Anda buat di akun Anda untuk mengizinkan layanan melakukan tindakan atas nama Anda. Peran layanan itu harus dimasukkan AWS Lambda sebagai kepala sekolah dalam kebijakan kepercayaan. Untuk detail tentang cara menggunakan kebijakan ini, lihat Cara Membuat Kebijakan AWS IAM untuk Memberikan AWS Lambda Akses ke Tabel AWS Amazon DynamoDB di Blog Keamanan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
```

```
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: Memungkinkan akses database RDS penuh dalam Wilayah tertentu

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan akses database RDS penuh dalam Wilayah tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:us-east-1:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
```

```
}
```

Amazon RDS: Memungkinkan memulihkan database RDS, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemulihan database RDS. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: Memungkinkan pemilik tag akses penuh ke sumber daya RDS yang telah mereka tag

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pemilik tag akses penuh ke sumber daya RDS yang telah mereka tag. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
```

```
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner":
 "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3: Memungkinkan pengguna Amazon Cognito mengakses objek di bucket mereka

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna Amazon Cognito mengakses objek di bucket Amazon S3 tertentu. Kebijakan ini mengizinkan akses hanya ke objek dengan nama yang menyertakan`cognito`, nama aplikasi, dan ID utama federasi, yang diwakili oleh variabel $ {cognito-identity.amazonaws.com:sub}. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

> **Note**
>
> Nilai 'sub' yang digunakan dalam kunci obyek bukanlah nilai sub pengguna di Kumpulan Pengguna, namun identitas id yang diasosiasikan dengan pengguna di Kumpulan Identitas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito menyediakan autentikasi, otorisasi, dan pengelolaan pengguna untuk aplikasi web dan seluler Anda. Pengguna Anda bisa langsung masuk dengan nama pengguna dan kata sandi, atau lewat pihak ketiga seperti Facebook, Amazon, atau Google.

Dua komponen utama Amazon Cognito adalah kumpulan pengguna dan kumpulan identitas. Kumpulan pengguna adalah direktori pengguna yang menyediakan opsi pendaftaran dan masuk bagi pengguna aplikasi Anda. Kumpulan identitas memungkinkan Anda memberi pengguna akses ke AWS layanan lain. Anda dapat menggunakan kolam identitas dan kolam pengguna secara terpisah atau bersama-sama.

Untuk informasi selengkapnya tentang Amazon Cognito, lihat Panduan Pengguna Amazon Cognito.

Amazon S3: Memungkinkan pengguna federasi mengakses ke direktori home Amazon S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan prinsipal federasi mengakses objek bucket direktori home mereka sendiri di S3. Direktori home adalah bucket yang menyertakan home folder dan folder untuk masing-masing kepala sekolah federasi. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *italicized placeholder text* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Variabel `${aws:userid}` dalam kebijakan ini memutuskan untuk `role-id:specified-name`. `role-id`Bagian dari ID utama federasi adalah pengidentifikasi unik yang ditugaskan untuk peran kepala sekolah federasi selama pembuatan. Untuk informasi selengkapnya, lihat Pengidentifikasi unik. `specified-name`Ini adalah RoleSessionName parameter yang diteruskan ke `AssumeRoleWithWebIdentity` permintaan ketika kepala sekolah federasi mengambil peran mereka.

Anda dapat melihat ID peran menggunakan AWS CLI perintah`aws iam get-role --role-name` *`specified-name`*. Misalnya, bayangkan Anda menentukan nama `John` yang mudah diingat dan CLI mengembalikan ID peran `AROAXXT2NJT7D3SIQN7Z6`. Dalam hal ini, ID pengguna kepala sekolah federasi adalah`AROAXXT2NJT7D3SIQN7Z6:John`. Kebijakan ini kemudian memungkinkan kepala sekolah federasi John untuk mengakses bucket Amazon S3 dengan awalan. `AROAXXT2NJT7D3SIQN7Z6:John`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3: Akses Bucket S3, tetapi bucket produksi ditolak tanpa MFA baru-baru ini

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan administrator Amazon S3 mengakses bucket apa pun, termasuk memperbarui, menambahkan, dan

menghapus objek. Namun, ini secara eksplisit menolak akses ke bucket `amzn-s3-demo-bucket-production` jika pengguna belum masuk menggunakan Autentikasi Multi-factor (MFA) dalam tiga puluh menit terakhir. Kebijakan ini memberikan izin yang diperlukan untuk melakukan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini tidak pernah mengizinkan akses terprogram ke bucket `amzn-s3-demo-bucket` menggunakan access key pengguna. Hal ini dicapai menggunakan kunci kondisi `aws:MultiFactorAuthAge` dengan operator kondisi `NumericGreaterThanIfExists`. Kondisi kebijakan ini akan kembali`true` jika tidak ada MFA atau jika usia MFA lebih dari 30 menit. Dalam situasi tersebut, akses ditolak. Untuk mengakses `amzn-s3-demo-bucket-production` bucket secara terprogram, administrator S3 harus menggunakan kredensial sementara yang dibuat dalam 30 menit terakhir menggunakan GetSessionToken operasi API.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
```

```
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-production/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3: Memungkinkan pengguna IAM mengakses direktori home S3 mereka, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengakses objek bucket direktori home mereka sendiri di S3. Direktori rumah adalah bucket yang mencakup home folder dan folders untuk pengguna individual. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Kebijakan ini tidak akan berfungsi saat menggunakan peran IAM karena `aws:username` variabel tidak tersedia saat menggunakan peran IAM. Untuk detail tentang nilai kunci prinsipal, lihat Nilai-nilai kunci utama.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: Batasi manajemen ke bucket S3 tertentu

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang membatasi pengelolaan bucket Amazon S3 ke bucket tertentu. Kebijakan ini memberikan izin untuk melakukan semua tindakan Amazon S3, tetapi menolak akses ke Layanan AWS setiap kecuali Amazon S3. Lihat contoh berikut ini. Menurut kebijakan ini, Anda hanya dapat mengakses tindakan Amazon S3 yang dapat Anda lakukan pada bucket S3 atau sumber daya objek S3. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Jika kebijakan ini digunakan dalam kombinasi dengan kebijakan lain (seperti kebijakan EC2 FullAccess AWS terkelola AmazonS3 atau FullAccess Amazon) yang mengizinkan tindakan ditolak oleh kebijakan ini, maka akses ditolak. Hal ini dikarenakan pernyataan penolakan yang eksplisit lebih diutamakan daripada pernyataan yang membolehkan. Untuk informasi selengkapnya, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.

> **Warning**
>
> `NotAction` dan `NotResource` adalah elemen kebijakan lanjutan yang harus digunakan dengan seksama. Kebijakan ini menolak akses ke setiap AWS layanan kecuali Amazon S3. Jika Anda melampirkan kebijakan ini kepada pengguna, kebijakan lain yang mengizinkan layanan lain diabaikan dan akses ditolak.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
```

```
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Berikan akses baca dan tulis ke objek bucket Amazon S3

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang mengizinkan `Read` dan `Write` mengakses objek di bucket Amazon S3 tertentu. Kebijakan ini memberikan izin yang diperlukan untuk menyelesaikan tindakan ini secara terprogram dari API atau. AWS AWS CLI Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Tindakan `s3:*Object` ini menggunakan wildcard sebagai bagian dari nama tindakan. Pernyataan ini `AllObjectActions` mengizinkan `GetObject`, `DeleteObject`, `PutObject`, dan tindakan Amazon S3 lainnya yang berakhir dengan kata “Obyek”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
```

```
    ]
}
```

> **Note**
>
> Untuk mengizinkan `Read` dan `Write` akses ke objek dalam bucket Amazon S3 dan juga termasuk izin tambahan untuk akses konsol, lihat Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol.

Amazon S3: Memungkinkan akses baca dan tulis ke objek di Bucket S3, secara terprogram dan di konsol

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang mengizinkan `Read` dan `Write` mengakses objek dalam bucket S3 tertentu. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol. Untuk menggunakan kebijakan ini, ganti kebijakan contoh *`italicized placeholder text`* dalam dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

Tindakan `s3:*Object` ini menggunakan wildcard sebagai bagian dari nama tindakan. Pernyataan ini `AllObjectActions` mengizinkan `GetObject`, `DeleteObject`, `PutObject`, dan tindakan Amazon S3 lainnya yang berakhir dengan kata “Obyek”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
        }
    ]
}
```

Kelola kebijakan IAM

IAM memberi Anda perangkat untuk membuat dan mengelola semua tipe kebijakan IAM (kebijakan terkelola dan kebijakan selaras). Untuk menambahkan izin ke identitas IAM (pengguna, grup, atau peran IAM), Anda membuat kebijakan, memvalidasi kebijakan, dan kemudian melampirkan kebijakan ke identitas. Anda dapat melampirkan beberapa kebijakan ke satu identitas, dan setiap kebijakan dapat berisi beberapa izin.

Topik

- Sumber daya tambahan
- Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan
- Validasi kebijakan IAM
- Pengujian kebijakan IAM dengan simulator kebijakan IAM
- Menambahkan dan menghapus izin identitas IAM
- Peningkatan versi IAM
- Edit kebijakan IAM
- Hapus kebijakan IAM
- Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda mempelajari lebih lanjut tentang AWS kebijakan.

- Untuk informasi lebih lanjut tentang tipe kebijakan IAM yang berbeda, lihat Kebijakan dan izin di AWS Identity and Access Management.
- Untuk informasi umum tentang penggunaan kebijakan di dalam IAM, lihat Manajemen akses untuk AWS sumber daya.
- Untuk informasi tentang cara menggunakan IAM Access Analyzer untuk menghasilkan kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas, lihat pembuatan kebijakan IAM Access Analyzer.
- Untuk informasi tentang bagaimana izin dievaluasi saat beberapa kebijakan berlaku untuk identitas IAM yang diberikan, lihat Logika evaluasi kebijakan.
- Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Lihat informasi yang lebih lengkap di IAM dan AWS STS kuota.

Tentukan izin IAM khusus dengan kebijakan terkelola pelanggan

Kebijakan menentukan izin untuk identitas atau sumber daya di. AWS Anda dapat membuat kebijakan yang dikelola pelanggan di IAM menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri Akun AWS. Anda kemudian dapat melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda Akun AWS.

Kebijakan berbasis identitas adalah kebijakan yang melekat pada identitas di IAM. Kebijakan berbasis identitas dapat mencakup kebijakan terkelola, kebijakan yang AWS dikelola pelanggan, dan kebijakan inline. AWS kebijakan terkelola dibuat dan dikelola oleh AWS, dan Anda dapat menggunakannya tetapi tidak mengelolanya. Kebijakan inline adalah kebijakan yang Anda buat dan sematkan langsung ke grup pengguna, pengguna, atau peran IAM. Kebijakan inline tidak dapat digunakan kembali pada identitas lain atau dikelola di luar identitas di mana mereka ada. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Umumnya lebih baik menggunakan kebijakan yang dikelola pelanggan daripada kebijakan inline atau kebijakan AWS terkelola. AWS kebijakan terkelola biasanya memberikan izin administratif atau hanya-baca yang luas. Untuk keamanan terbesar, berikan hak istimewa paling sedikit, yang berarti hanya memberikan izin yang diperlukan untuk melakukan tugas pekerjaan tertentu.

Ketika Anda membuat atau mengedit kebijakan IAM, AWS dapat secara otomatis melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif dengan hak istimewa paling sedikit dalam pikiran. Dalam Konsol Manajemen AWS, IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan Anda lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Anda dapat menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API untuk membuat kebijakan yang dikelola pelanggan di IAM. Untuk informasi selengkapnya tentang menggunakan CloudFormation templat untuk menambah atau memperbarui kebijakan, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan CloudFormation Pengguna.

Topik

- Buat kebijakan IAM (konsol)
- Buat kebijakan IAM ()AWS CLI
- Buat kebijakan IAM (AWS API)

Buat kebijakan IAM (konsol)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan Konsol Manajemen AWS untuk membuat kebijakan yang dikelola pelanggan di IAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri. Akun AWS Anda kemudian dapat melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda Akun AWS.

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Topik

- Membuat kebijakan IAM
- Membuat kebijakan menggunakan editor JSON
- Membuat kebijakan dengan editor visual
- Mengimpor kebijakan terkelola yang ada

Membuat kebijakan IAM

Anda dapat membuat kebijakan terkelola pelanggan Konsol Manajemen AWS menggunakan salah satu metode berikut:

- JSON — Tempel dan sesuaikan publikasi contoh kebijakan berbasis-identitas.
- Editor visual — Susun kebijakan baru dari nol di editor visual. Jika Anda menggunakan editor visual, Anda tidak harus memahami sintaksis JSON.
- Impor — Impor dan sesuaikan kebijakan terkelola dari akun Anda. Anda dapat mengimpor kebijakan AWS terkelola atau kebijakan terkelola pelanggan yang sebelumnya Anda buat.

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Membuat kebijakan menggunakan editor JSON

Anda dapat mengetik atau menempelkan kebijakan di JSON dengan memilih opsi JSON. Metode ini berguna untuk menyalin contoh kebijakan untuk dipakai di akun Anda. Atau, Anda bisa mengetik dokumen kebijakan JSON Anda sendiri di editor JSON. Anda juga dapat menggunakan opsi JSON untuk beralih antara editor visual dan JSON untuk membandingkan tampilan.

Saat Anda membuat atau mengedit kebijakan di editor JSON, IAM melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda menyempurnakan kebijakan lebih lanjut.

Dokumen kebijakan JSON terdiri atas satu atau lebih pernyataan. Setiap pernyataan harus berisi semua tindakan yang berbagi efek yang sama (`Allow` atau `Deny`) dan mendukung sumber daya dan kondisi yang sama. Jika satu tindakan mengharuskan Anda untuk menentukan semua sumber daya (`"*"`) dan tindakan lain mendukung Amazon Resource Name (ARN) dari sumber daya tertentu, mereka harus berada dalam dua pernyataan JSON terpisah. Untuk rincian format ARN, lihat Amazon Resource Name (ARN) di dalam Panduan Referensi Umum AWS . Untuk informasi umum tentang kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management. Untuk informasi tentang bahasa kebijakan IAM, lihat Referensi kebijakan IAM JSON.

Cara menggunakan editor kebijakan JSON untuk membuat kebijakan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel dokumen kebijakan JSON. Untuk rincian bahasa kebijakan IAM, lihat Referensi kebijakan IAM JSON.
6. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

   > **Note**
   >
   > Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

7. (Opsional) Saat membuat atau mengedit kebijakan Konsol Manajemen AWS, Anda dapat membuat templat kebijakan JSON atau YAMAL yang dapat Anda gunakan dalam CloudFormation templat.

   Untuk melakukannya, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya, CloudFormation lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.
8. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
9. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda.
10. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
11. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Membuat kebijakan dengan editor visual

Editor visual di konsol IAM akan memandu Anda dalam membuat kebijakan tanpa harus menulis sintaksis JSON. Untuk melihat contoh penggunaan editor visual dalam pembuatan kebijakan, lihat the section called “Mengontrol akses ke identitas”.

Untuk menggunakan editor visual dalam pembuatan kebijakan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, temukan bagian Pilih layanan, lalu pilih AWS layanan. Anda bisa menggunakan kotak pencarian di bagian atas untuk membatasi hasil pada daftar layanan. Anda bisa memilih hanya satu layanan pada blok izin editor visual. Untuk memberikan akses ke lebih dari satu layanan, tambahkan beberapa blok izin dengan memilih Tambahkan lebih banyak izin.
5. Di Tindakan yang diizinkan, pilih tindakan yang akan ditambahkan ke kebijakan. Anda bisa memilih tindakan dengan cara berikut:

   - Pilih kotak centang untuk semua tindakan.
   - Pilih tambah tindakan untuk mengetik nama tindakan tertentu. Anda bisa menggunakan wildcards (*) untuk menentukan beberapa tindakan.
   - Pilih satu grup Tingkat akses untuk memilih semua tindakan untuk tingkat akses tersebut (misalnya, Baca, Tulis, atau Daftar).
   - Perluas setiap grup Tingkat akses untuk memilih tindakan individu.

   Secara default, kebijakan yang Anda buat mengizinkan tindakan yang Anda pilih. Sebaliknya, untuk menolak tindakan terpilih, pilih Beralih ke menolak izin. Karena IAM menolak secara default, kami merekomendasikan sebagai praktik terbaik keamanan agar Anda mengizinkan hanya tindakan dan sumber daya yang diperlukan pengguna saja. Anda harus membuat pernyataan JSON untuk menolak izin hanya bila Anda ingin membatalkan izin secara terpisah mengizinkan pernyataan atau kebijakan lain. Kami sarankan Anda membatasi jumlah izin penolakan seminim mungkin karena dapat meningkatkan kesulitan izin pemecahan masalah.

6. Untuk Sumber Daya, bila layanan dan tindakan yang Anda pilih di langkah sebelumnya tidak mendukung pilihan sumber daya tertentu, semua sumber daya diperbolehkan dan Anda tidak bisa mengedit bagian ini.

   Jika Anda memilih satu atau lebih tindakan yang mendukungizin tingkat sumber daya, maka editor visual akan mendaftar sumber daya tersebut. Kemudian Anda bisa memperluas Sumber Daya untuk menentukan sumber daya bagi kebijakan Anda.

   Anda dapat menentukan sumber daya dengan cara berikut:

   - Pilih Tambah ARNs untuk menentukan sumber daya berdasarkan Nama Sumber Daya Amazon (ARN) mereka. Anda dapat menggunakan editor atau daftar ARNs ARN visual secara manual. Untuk informasi lebih lanjut tentang sintaks ARN, lihat Amazon Resource Name (ARN) pada Panduan Referensi Umum AWS . Untuk informasi tentang penggunaan ARNs dalam `Resource` elemen kebijakan, lihatElemen kebijakan IAM JSON: Sumber daya.
   - Pilih Apa saja di akun ini di samping sumber daya untuk memberikan izin ke sumber daya apa pun dari jenis itu.
   - Pilih Semua untuk memilih semua sumber daya untuk layanan ini.

7. (Opsional) Pilih Ketentuan permintaan - opsional untuk menambahkan kondisi ke kebijakan yang Anda buat. Kondisi membatasi efek dari pernyataan kebijakan JSON. Misalnya, Anda dapat menentukan bahwa pengguna diizinkan melakukan tindakan pada sumber daya hanya ketika permintaan pengguna tersebut terjadi di rentang waktu tertentu. Anda juga bisa menggunakan kondisi yang umum dipakai untuk membatasi apakah seorang pengguna harus menggunakan multi-factor authentication (MFA). Atau Anda bisa meminta agar permintaan yang berasal dari rentang alamat IP tertentu. Untuk daftar semua kunci konteks yang dapat Anda gunakan dalam kondisi kebijakan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

   Anda bisa memilih kondisi dengan cara berikut:

   - Gunakan kotak centang untuk memilih kondisi yang umum digunakan.
   - Pilih Tambahkan kondisi lain untuk menentukan kondisi lain. Pilih kondisi dari Kunci Kondisi, Pengukur, dan Operator, lalu ketik Nilai. Untuk menambahkan lebih dari satu nilai, pilih Tambah. Anda dapat mempertimbangkan nilai tersebut terhubung secara logika "ATAU" operator. Setelah selesai, pilih Tambahkan kondisi.

Untuk menambahkan lebih dari satu kondisi, pilih Tambahkan kondisi lain lagi. Ulangi seperlunya. Setiap kondisi berlaku hanya untuk blok izin editor visual yang satu ini. Semua kondisi haruslah benar agar blok izin dapat dianggap cocok. Dengan kata lain, pertimbangkan kondisi yang akan dihubungkan oleh logika “DAN” operator.

Untuk informasi lebih lanjut mengenai elemen Kondisi, lihat Elemen kebijakan IAM JSON: Kondisi di Referensi kebijakan IAM JSON.

8. Untuk menambahkan lebih banyak blok izin, pilih Tambahkan izin lainnya. Untuk setiap blok, ulangi langkah 2 sampai 5.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

9. (Opsional) Saat membuat atau mengedit kebijakan Konsol Manajemen AWS, Anda dapat membuat templat kebijakan JSON atau YAMAL yang dapat Anda gunakan dalam CloudFormation templat.

   Untuk melakukannya, di editor Kebijakan pilih Tindakan, lalu pilih Buat CloudFormation templat. Untuk mempelajari selengkapnya, CloudFormation lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan AWS CloudFormation Pengguna.

10. Setelah selesai menambahkan izin ke kebijakan, pilih Berikutnya.
11. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk memastikan bahwa Anda telah memberikan izin yang dimaksud.
12. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang menggunakan tanda di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya.
13. Pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.

Mengimpor kebijakan terkelola yang ada

Cara mudah untuk membuat kebijakan baru adalah dengan mengimpor kebijakan terkelola yang ada di dalam akun Anda yang setidaknya memiliki beberapa izin yang Anda perlukan. Anda kemudian bisa mengubah kebijakan tersebut untuk menyesuaikannya dengan persyaratan baru Anda.

Anda tidak bisa mengimpor kebijakan inline. Untuk mempelajari perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Untuk mengimpor kebijakan terkelola yang sudah ada di editor visual

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di editor Kebijakan, pilih Visual dan kemudian di sisi kanan halaman, pilih Tindakan lalu pilih Kebijakan impor.
5. Di jendela Kebijakan impor, pilih kebijakan terkelola yang paling cocok dengan kebijakan yang ingin Anda sertakan dalam kebijakan baru Anda. Anda dapat menggunakan kotak pencarian di bagian atas untuk membatasi hasil dalam daftar kebijakan.
6. Pilih kebijakan Impor.

   Kebijakan yang diimpor ditambahkan pada blok izin baru di bagian bawah kebijakan Anda.
7. Gunakan editor Visual atau pilih JSON untuk menyesuaikan kebijakan Anda. Lalu pilih Selanjutnya.

   > **Note**
   >
   > Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.
8. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Anda tidak bisa mengedit pengaturan kemudian. Tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Untuk mengimpor kebijakan terkelola yang ada di editor JSON

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON, lalu di sisi kanan halaman, pilih Tindakan lalu pilih Kebijakan impor.
5. Di jendela Kebijakan impor, pilih kebijakan terkelola yang paling cocok dengan kebijakan yang ingin Anda sertakan dalam kebijakan baru Anda. Anda dapat menggunakan kotak pencarian di bagian atas untuk membatasi hasil dalam daftar kebijakan.
6. Pilih kebijakan Impor.

   Pernyataan dari kebijakan yang diimpor ditambahkan di bagian bawah kebijakan JSON Anda.
7. Sesuaikan kebijakan Anda di JSON. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Sesuaikan kebijakan Anda di JSON, atau pilih Visual editor (Editor visual). Lalu pilih Selanjutnya.

   > **Note**
   >
   > Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.
8. Pada halaman Tinjau dan buat, ketik Nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Anda tidak bisa mengedit hal ini kemudian. Tinjau kebijakan Izin yang ditentukan dalam kebijakan ini, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Setelah Anda membuat kebijakan, Anda dapat melampirkannya ke grup, pengguna, atau peran Anda. Lihat informasi yang lebih lengkap di Menambahkan dan menghapus izin identitas IAM.

Buat kebijakan IAM ()AWS CLI

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS CLI untuk membuat kebijakan yang dikelola pelanggan di IAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola

sendiri. Akun AWS Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional. Dengan memvalidasi kebijakan Anda, Anda dapat mengatasi kesalahan atau rekomendasi apa pun sebelum Anda melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda. Akun AWS

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Membuat kebijakan IAM ()AWS CLI

Anda dapat membuat kebijakan terkelola pelanggan IAM atau kebijakan inline menggunakan AWS Command Line Interface ()AWS CLI.

Untuk membuat kebijakan terkelola pelanggan (AWS CLI)

Gunakan perintah berikut ini:

- buat-kebijakan

Untuk membuat kebijakan inline untuk identitas IAM (grup, pengguna, atau peran) ()AWS CLI

Gunakan salah satu perintah berikut:

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Anda tidak dapat menggunakan IAM untuk menanamkan kebijakan yang selaras untuk peran terkait-layanan.

Untuk memvalidasi kebijakan terkelola pelanggan (AWS CLI)

Gunakan perintah IAM Access Analyzer berikut:

- validasi-kebijakan

Buat kebijakan IAM (AWS API)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS API untuk membuat kebijakan terkelola pelanggan di IAM. Kebijakan yang dikelola pelanggan adalah kebijakan mandiri yang Anda kelola sendiri. Akun AWS Sebagai praktik terbaik, kami menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional. Dengan memvalidasi kebijakan Anda, Anda dapat mengatasi kesalahan atau rekomendasi apa pun sebelum Anda melampirkan kebijakan ke identitas (pengguna, grup, dan peran) di situs Anda. Akun AWS

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Membuat kebijakan IAM (AWS API)

Anda dapat membuat kebijakan terkelola pelanggan IAM atau kebijakan inline menggunakan API. AWS

Untuk membuat kebijakan terkelola pelanggan (AWS API)

Hubungi operasi berikut ini:

- CreatePolicy

Untuk membuat kebijakan inline untuk identitas IAM (grup, pengguna, atau peran) (AWS API)

Panggil salah satu operasi berikut ini:

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Anda tidak dapat menggunakan IAM untuk menanamkan kebijakan yang selaras untuk peran terkait-layanan.

Untuk memvalidasi kebijakan terkelola pelanggan (AWS API)

Hubungi operasi IAM Access Analyzer berikut:

- ValidatePolicy

Validasi kebijakan IAM

Kebijakan adalah dokumen JSON yang menggunakan tata bahasa kebijakan IAM. Saat Anda melampirkan kebijakan pada entitas IAM, seperti pengguna, grup, atau peran, ia mengizinkan entitas tersebut.

Saat Anda membuat atau mengedit kebijakan kontrol akses IAM menggunakan Konsol Manajemen AWS, secara AWS otomatis memeriksanya untuk memastikan bahwa kebijakan tersebut mematuhi tata bahasa kebijakan IAM. Jika AWS menentukan sebuah kebijakan tidak sesuai dengan tata bahasa, ia akan meminta Anda memperbaiki kebijakan tersebut.

IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh IAM Access Analyzer, lihat referensi pemeriksaan kebijakan IAM Access Analyzer.

Cakupan validasi

AWS memeriksa sintaks kebijakan JSON dan tata bahasa. Ini juga memverifikasi bahwa Anda ARNs diformat dengan benar dan nama tindakan dan kunci kondisi sudah benar.

Mengakses validasi kebijakan

Kebijakan divalidasi secara otomatis saat Anda membuat kebijakan JSON atau mengedit kebijakan yang ada di. Konsol Manajemen AWS Jika sintaks kebijakan tidak valid, Anda menerima notifikasi dan harus memperbaiki masalahnya sebelum bisa melanjutkan. Temuan dari validasi kebijakan IAM Access Analyzer secara otomatis dikembalikan dalam Konsol Manajemen AWS jika Anda memiliki izin untuk. `access-analyzer:ValidatePolicy` Anda juga dapat memvalidasi kebijakan menggunakan AWS API atau AWS CLI.

Kebijakan yang sudah Ada

Anda mungkin memiliki kebijakan yang sudah ada yang tidak valid karena dibuat atau terakhir disimpan sebelum pembaruan terbaru ke mesin kebijakan. Sebagai praktik terbaik, kami

menyarankan Anda menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional. Kami menyarankan Anda membuka kebijakan yang ada dan meninjau hasil validasi kebijakan yang dihasilkan. Anda tidak dapat mengedit dan menyimpan kebijakan yang sudah ada tanpa memperbaiki kesalahan sintaks kebijakan apa pun.

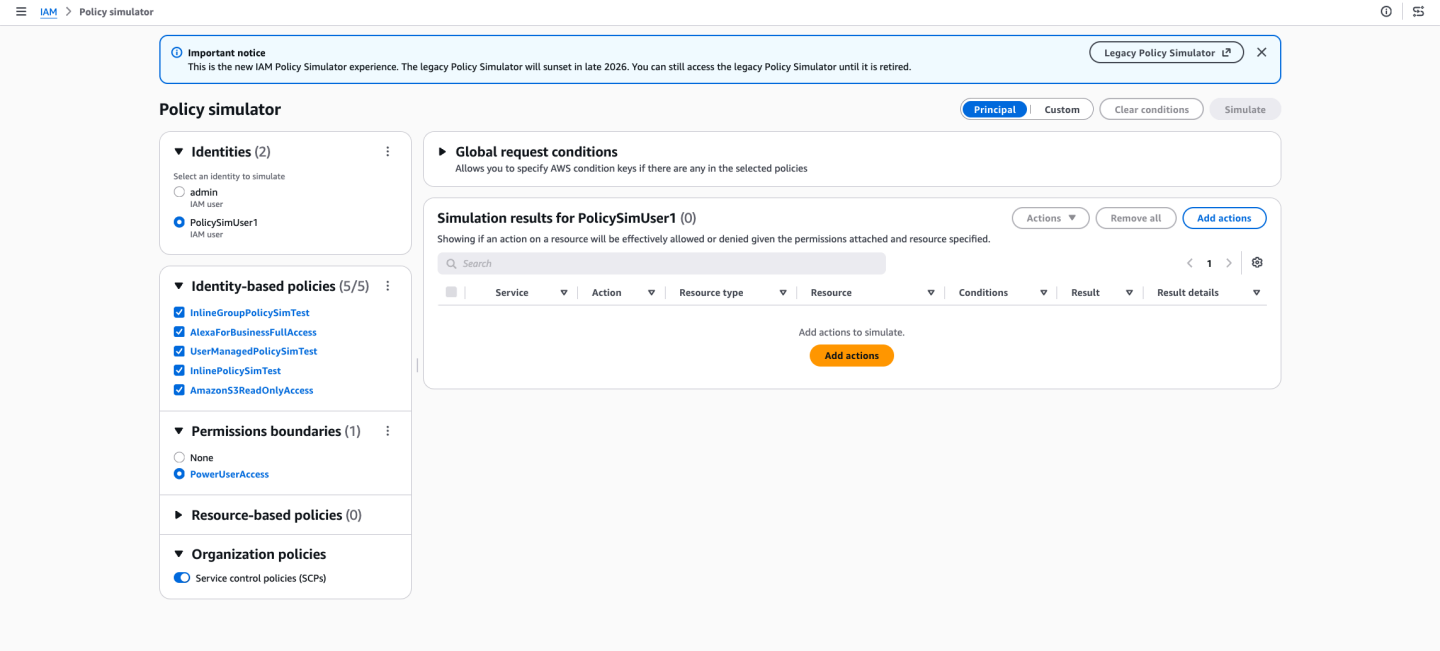
Pengujian kebijakan IAM dengan simulator kebijakan IAM

Dengan simulator kebijakan IAM, Anda dapat menguji kebijakan berbasis identitas, batas izin IAM, kebijakan kontrol layanan (SCP), dan kebijakan berbasis sumber daya yang Anda berikan. Anda mensimulasikan bagaimana AWS akan mengevaluasi permintaan terhadap kebijakan tersebut, tanpa mengirim permintaan nyata ke AWS layanan apa pun. Setiap simulasi mengembalikan hasil otorisasi biner untuk setiap tindakan dan sumber daya yang Anda uji: tindakan diizinkan atau ditolak. Ketika hasilnya adalah izin atau penolakan eksplisit, simulator juga menunjukkan kepada Anda kebijakan mana yang menghasilkan hasil itu, sehingga Anda dapat memahami dan memperbaiki penyebabnya.

Untuk informasi lebih lanjut tentang bagaimana dan mengapa menggunakan kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management.

> **Important**
>
> Hasil simulator kebijakan dapat berbeda dari AWS lingkungan hidup Anda. Kami menyarankan Anda memeriksa kebijakan terhadap AWS lingkungan langsung Anda setelah pengujian dengan simulator kebijakan untuk mengonfirmasi bahwa Anda mendapatkan hasil yang diinginkan. Untuk informasi selengkapnya, lihat Bagaimana simulator kebijakan IAM bekerja.

Tangkapan layar berikut menunjukkan simulator kebijakan IAM, di mana Anda memilih identitas dan kebijakannya dan memilih tindakan dan sumber daya untuk diuji sebelum menjalankan simulasi.



Topik

- Mode simulator kebijakan
- Apa yang dapat Anda lakukan dengan simulator kebijakan
- Bagaimana simulator kebijakan IAM bekerja
- Cara kerja evaluasi kunci kondisi di simulator kebijakan IAM
- Izin untuk menggunakan simulator kebijakan
- Cara mensimulasikan kebijakan

Mode simulator kebijakan

Simulator kebijakan memiliki dua mode. Pilih mode yang cocok apakah Anda ingin menguji kebijakan yang sudah ada di akun Anda, atau kebijakan yang masih Anda susun.

Modus utama

Dalam mode Principal, Anda menguji kebijakan yang sudah dilampirkan ke pengguna, peran, atau grup IAM yang ada. Anda memilih satu identitas untuk simulasi, dan simulator mengevaluasi kebijakan yang melekat pada identitas itu. Anda juga dapat menyertakan atau mengecualikan kebijakan identitas kustom atau batas izin ke simulasi. Penambahan tersebut disimulasikan seolah-olah dilampirkan ke prinsipal, tanpa benar-benar dilampirkan di IAM, sehingga akun Anda tidak berubah.

Modus kustom

Dalam mode Kustom, Anda menguji satu atau beberapa kebijakan yang Anda tulis atau tempel. Ini berguna untuk memeriksa kebijakan sebelum Anda melampirkannya. Kebijakan khusus disimulasikan sendiri dan tidak disimpan ke akun Anda.

Untuk petunjuk langkah demi langkah terperinci tentang memilih mode dan menjalankan simulasi, lihatCara mensimulasikan kebijakan.

Apa yang dapat Anda lakukan dengan simulator kebijakan

Berikut adalah beberapa hal umum yang dapat Anda lakukan dengan simulator kebijakan. Panduan langkah demi langkah yang lebih rinci dan perbedaan antara perilaku API dan konsol dapat ditemukan diCara mensimulasikan kebijakan.

- Uji kebijakan berbasis identitas, apakah sudah dilampirkan ke pengguna, grup, atau peran IAM, atau kebijakan baru yang Anda tulis atau tempel. Untuk identitas terlampir dengan lebih dari satu kebijakan, Anda dapat menguji semua kebijakan bersama-sama atau memilih kebijakan individual, dan melihat tindakan mana yang diizinkan atau ditolak untuk sumber daya tertentu. Kebijakan baru dievaluasi seolah-olah dilampirkan ke identitas, tanpa benar-benar melampirkannya di akun Anda; mereka hanya digunakan dalam simulasi dan tidak disimpan ke akun Anda.
- Uji efek kontrol ketat pada akses. Anda dapat mensimulasikan satu batas izin pada satu waktu. Jika Anda Akun AWS adalah anggota organisasi di AWS Organizations, Anda juga dapat menguji dampak kebijakan kontrol layanan (SCP) pada kebijakan berbasis identitas Anda. Simulator memberi tahu Anda apakah SCP mengizinkan atau menolak tindakan, tetapi untuk alasan keamanan, itu tidak menampilkan pernyataan yang cocok dari SCP seperti yang dilakukan untuk jenis kebijakan lainnya.
- Uji efek kebijakan berbasis sumber daya yang Anda berikan. Kecuali di konsol, simulator tidak mengambil kebijakan sumber daya untuk Anda. Untuk menyertakan kebijakan berbasis sumber daya dalam simulasi, Anda memberikan kebijakan bersama dengan sumber daya.
- Uji layanan, tindakan, dan sumber daya tertentu dalam kondisi dunia nyata, dan identifikasi apa yang menentukan hasilnya. Misalnya, Anda dapat menguji apakah kebijakan mengizinkan`ListAllMyBuckets`,`CreateBucket`, dan `DeleteBucket` tindakan di Amazon S3 pada bucket tertentu, dan memberikan kunci konteks seperti alamat IP atau tanggal yang direferensikan dalam `Condition` elemen dalam kebijakan yang sedang diuji. Untuk setiap hasil, simulator mengidentifikasi pernyataan spesifik mana yang mengizinkan atau menolak akses; pernyataan yang cocok dapat berasal dari kebijakan berbasis identitas dan dari kebijakan berbasis sumber daya yang Anda berikan.

Bagaimana simulator kebijakan IAM bekerja

Saat Anda menjalankan simulasi, simulator kebijakan mengevaluasi kebijakan dalam cakupan permintaan, SCP apa pun dalam cakupan, dan input yang Anda berikan, seperti tindakan, sumber daya, dan nilai kunci konteks. Dalam mode Kustom Anda memberikan kebijakan secara langsung. Dalam mode Principal simulator menggunakan kebijakan yang dilampirkan pada identitas yang Anda pilih, dan Anda dapat menyertakan atau mengecualikan lebih banyak kebijakan ke simulasi. Hasilnya adalah hasil biner untuk setiap tindakan dan sumber daya: diizinkan atau ditolak.

Simulator kebijakan berbeda dari AWS lingkungan hidup dengan cara berikut:

- Simulator kebijakan tidak membuat permintaan AWS layanan yang sebenarnya, sehingga Anda dapat dengan aman menguji permintaan yang mungkin membuat perubahan yang tidak diinginkan pada AWS lingkungan langsung Anda. Itu tidak menggunakan nilai kunci konteks nyata dari produksi, dan karena tindakan tidak benar-benar dijalankan, ia tidak mengembalikan respons layanan. Satu-satunya hasil adalah apakah setiap tindakan yang diminta akan diizinkan atau ditolak.
- Jika Anda mengedit kebijakan di simulator kebijakan, perubahan hanya memengaruhi simulasi. Kebijakan yang sesuai di Anda Akun AWS tetap tidak berubah.
- Simulator kebijakan mengevaluasi SCP, termasuk kunci kondisi dan cakupan sumber daya dalam pernyataan menolak, tetapi tidak mendukung kebijakan kontrol sumber daya (RCP). Hasil simulasi masih dapat berbeda dari perilaku langsung untuk beberapa konfigurasi lanjutan, seperti kebijakan titik akhir VPC, rantai peran, atau beberapa kebijakan berbasis sumber daya pada satu sumber daya.
- Cross-account simulasi mengembalikan keputusan tipe-kebijakan (kebijakan identitas dan kebijakan sumber daya) di. `EvalDecisionDetails`

Cara kerja evaluasi kunci kondisi di simulator kebijakan IAM

Banyak kebijakan menggunakan `Condition` elemen yang bergantung pada kunci kondisi, seperti `aws:RequestedRegion` atau`aws:MultiFactorAuthPresent`. Saat Anda menjalankan simulasi, simulator kebijakan mendapatkan nilai untuk kunci kondisi dari dua sumber: nilai yang diisi simulator secara otomatis, dan nilai yang Anda berikan sebagai input simulasi. Jika suatu nilai tidak memenuhi suatu kondisi, tindakan tersebut ditolak.

Konteks yang diisi simulator secara otomatis

Simulator mengisi kunci konteks prinsip dan organisasi berikut untuk evaluasi:

- Konteks utama: `aws:PrincipalAccountaws:PrincipalId`,`aws:PrincipalType`,`aws:Type`,`aws:UserId`, `aws:UserName`
- Konteks organisasi:`aws:PrincipalOrgID`,`aws:PrincipalOrgMasterAccountId`, `aws:PrincipalOrgPaths`

Karena kunci ini diisi untuk Anda, SCP dan kebijakan lain yang menggunakan kondisi prinsip atau organisasi dievaluasi dengan benar. Anda memberikan nilai untuk semua kunci kondisi lain yang dirujuk kebijakan Anda. Tinjau `Condition` elemen dalam kebijakan yang Anda uji dan berikan nilai untuk kunci apa pun yang tidak ada dalam daftar di atas.

Kunci kondisi global dan dukungan layanan

Untuk kebijakan berbasis identitas dan kebijakan berbasis sumber daya, simulator tidak menentukan layanan mana yang mendukung kunci kondisi global tertentu untuk otorisasi. Misalnya, simulator tidak mengidentifikasi bahwa layanan tidak mendukung`aws:TagKeys`. Sebaliknya, kebijakan kontrol layanan (SCP) dievaluasi dengan logika sadar layanan, termasuk kunci kondisi global seperti dan. `aws:RequestedRegion` `aws:PrincipalAccount` Untuk informasi lebih lanjut tentang kunci syarat ini, lihat AWS kunci konteks kondisi global.

Kunci kondisi dalam kebijakan kontrol layanan (SCP)

Simulator mengevaluasi kunci kondisi dan pelingkupan sumber daya di SCP. Bagaimana Anda memberikan nilai untuk kunci tersebut tergantung pada cara Anda menjalankan simulasi:

- Dengan AWS API AWS CLI or, Anda dapat memberikan nilai untuk kunci kondisi yang direferensikan oleh SCP.
- Di konsol IAM, Anda hanya dapat menetapkan nilai untuk kunci kondisi yang muncul di kebijakan berbasis identitas, batas izin, atau kebijakan berbasis sumber daya. Jika kunci kondisi direferensikan hanya oleh SCP, Anda tidak dapat menetapkan nilainya di konsol. Namun, jika kunci kondisi yang sama juga muncul di salah satu kebijakan lain tersebut, nilai yang Anda tetapkan digunakan di seluruh evaluasi, termasuk SCP.

Untuk alasan keamanan, evaluasi SCP tidak mengembalikan nilai konteks yang hilang. Anda masih menerima keputusan izinkan atau menolak untuk permintaan tersebut.

Izin untuk menggunakan simulator kebijakan

Izin yang Anda butuhkan tergantung pada apakah Anda menggunakan AWS API AWS CLI atau, atau konsol IAM, dan mode mana yang Anda gunakan.

Console

Anda memerlukan izin untuk menelepon `iam:GetContextKeysForCustomPolicy` untuk menggunakan simulator kebijakan IAM di konsol.

Izin tambahan yang Anda butuhkan tergantung pada mode. Dalam mode Principal, Anda memerlukan izin baca tambahan untuk mencantumkan identitas di akun Anda dan membaca kebijakan terlampir. Untuk mengurangi jumlah yang harus Anda berikan, gunakan mode Kustom jika Anda hanya perlu menguji kebijakan yang Anda tempelkan.

Modus kustom

Untuk menggunakan mode Kustom, di mana Anda menempelkan kebijakan yang ingin Anda uji, Anda juga perlu `iam:SimulateCustomPolicy`

Modus utama

Untuk menggunakan mode Principal, di mana konsol memuat dan menampilkan identitas dan kebijakan di akun Anda, Anda memerlukan izin umum ditambah yang berikut ini. Ini adalah set yang lebih besar, karena konsol harus menemukan identitas dan membaca kebijakan terlampir.

Simulasikan:

- `iam:SimulatePrincipalPolicy`

Identitas daftar:

- `iam:ListUsers`, `iam:ListRoles`, `iam:ListGroups`
- `iam:ListGroupsForUser`

Daftar kebijakan terlampir:

- `iam:ListPolicies`, `iam:ListUserPolicies`, `iam:ListRolePolicies`, `iam:ListGroupPolicies`

- `iam:ListAttachedUserPolicies`, `iam:ListAttachedRolePolicies`, `iam:ListAttachedGroupPolicies`

Mendapatkan dokumen kebijakan:

- `iam:GetPolicyVersion`, `iam:GetUserPolicy`, `iam:GetRolePolicy`, `iam:GetGroupPolicy`

Mendapatkan batas izin:

- `iam:GetUser`, `iam:GetRole`

Izin tambahan untuk mode Principal untuk mensimulasikan kebijakan berbasis sumber daya

Untuk menyertakan kebijakan sumber daya dalam simulasi mode Principal, tambahkan izin baca untuk jenis sumber daya tersebut:

- `s3:GetBucketPolicy`, `sns:GetTopicAttributes`, `sqs:GetQueueUrl`, `sqs:GetQueueAttributes`, `glacier:GetVaultAccessPolicy`

Misalnya kebijakan yang memberikan akses ke simulator, lihatthe section called “Contoh kebijakan: AWS Identity and Access Management (IAM)”.

AWS CLI and AWS API

Untuk AWS API AWS CLI and, Anda hanya perlu izin untuk memanggil tindakan simulasi untuk jenis kebijakan yang ingin Anda uji. Jika Anda menggunakan `iam:GetContextKeysForCustomPolicy` dan `iam:GetContextKeysForPrincipalPolicy` menemukan kunci konteks terlebih dahulu, berikan juga izin untuk tindakan ini. Untuk alasan keamanan, tindakan dibagi menjadi dua mode berikut. Berikan hanya izin yang cocok dengan apa yang ingin Anda simulasikan.

Untuk mensimulasikan kebijakan yang Anda berikan langsung ke API sebagai string (kebijakan khusus)

- `iam:SimulateCustomPolicy`
- `iam:GetContextKeysForCustomPolicy`(hanya jika Anda menggunakan langkah penemuan kunci konteks)

Tindakan ini memungkinkan Anda menguji kebijakan yang belum dilampirkan ke pengguna, grup pengguna, atau peran. Anda meneruskan kebijakan sebagai string, dan mereka hanya digunakan dalam simulasi.

Untuk mensimulasikan kebijakan yang dilampirkan ke pengguna IAM, grup pengguna, peran, atau sumber daya

- `iam:SimulatePrincipalPolicy`
- `iam:GetContextKeysForPrincipalPolicy`(hanya jika Anda menggunakan langkah penemuan kunci konteks)

Karena tindakan ini dapat mengungkapkan izin yang ditetapkan ke entitas IAM lainnya, pertimbangkan untuk membatasi siapa yang dapat memanggilnya. Misalnya, untuk memberi pengguna bernama Bob izin untuk mensimulasikan kebijakan yang ditetapkan untuk pengguna bernama Alice, berikan Bob akses ke sumber daya`arn:aws:iam::777788889999:user/alice`.

Untuk melihat contoh kebijakan yang memungkinkan penggunaan API simulator kebijakan untuk kebijakan terlampir dan tidak dilampirkan saat ini Akun AWS, lihatIAM: Akses API simulator kebijakan. Untuk melihat contoh kebijakan yang dicakup oleh pengguna dengan jalur tertentu, lihatIAM: Akses API simulator kebijakan berdasarkan jalur pengguna.

Cara mensimulasikan kebijakan

Anda dapat menjalankan simulasi di konsol IAM atau dengan AWS API AWS CLI dan. Untuk ikhtisar tentang apa yang dievaluasi simulator, dua mode, dan bagaimana mencapai hasil, lihatPengujian kebijakan IAM dengan simulator kebijakan IAM. Untuk izin yang Anda butuhkan, lihatIzin untuk menggunakan simulator kebijakan.

Topik

- Simulasikan kebijakan (konsol)
- Simulasikan kebijakan (AWS CLI and AWS API)

Simulasikan kebijakan (konsol)

Anda dapat menjalankan simulasi dari simulator kebijakan di konsol https://console.aws.amazon.com/iam/ IAM. Di panel navigasi, pilih simulator kebijakan.

> **Note**
>
> Konsol simulator kebijakan mandiri di https://policysim.aws.amazon.com/ tidak lagi dipertahankan. Gunakan simulator kebijakan di konsol IAM.

Modus utama

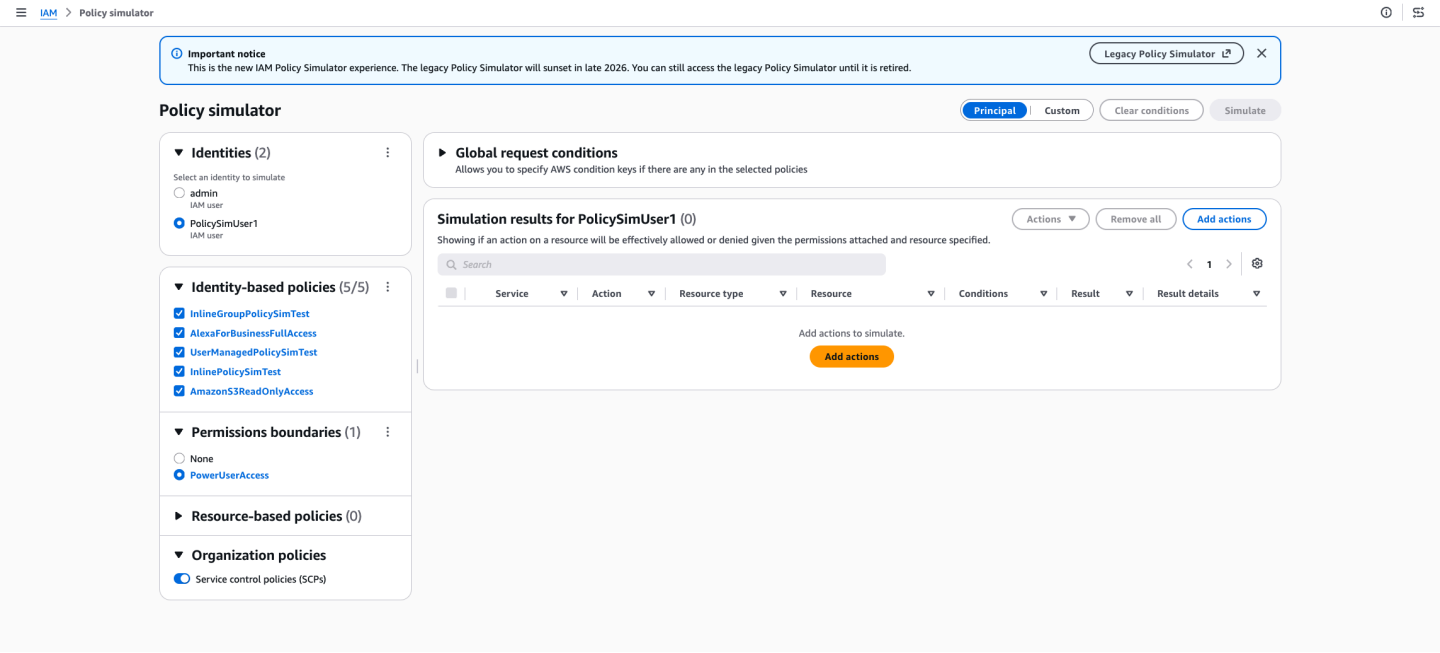
Pilih identitas

Pertama-tama Anda mulai dengan memilih identitas. Anda dapat memilih pengguna, peran, atau grup IAM di akun tersebut. Panel samping akan mengisi kebijakan yang dilampirkan pada identitas ini setelah pilihan Anda.

Sertakan atau mengecualikan kebijakan

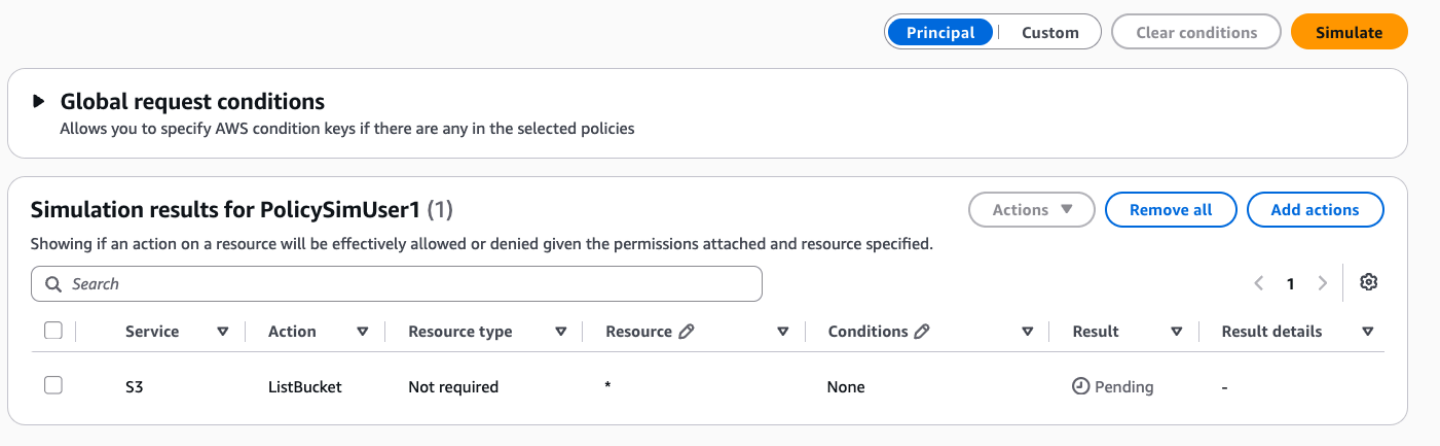
Panel samping mencantumkan kebijakan dalam lingkup simulasi, termasuk kebijakan sebaris, ter AWS kelola, dan dikelola pelanggan, dan batas izin apa pun. Anda dapat mengecualikan kebijakan untuk melihat bagaimana hasilnya berubah tanpanya, lalu memasukkannya lagi. Mengecualikan kebijakan yang memberikan akses mengubah hasilnya menjadi penolakan implisit.

Tambahkan tindakan dan sumber daya

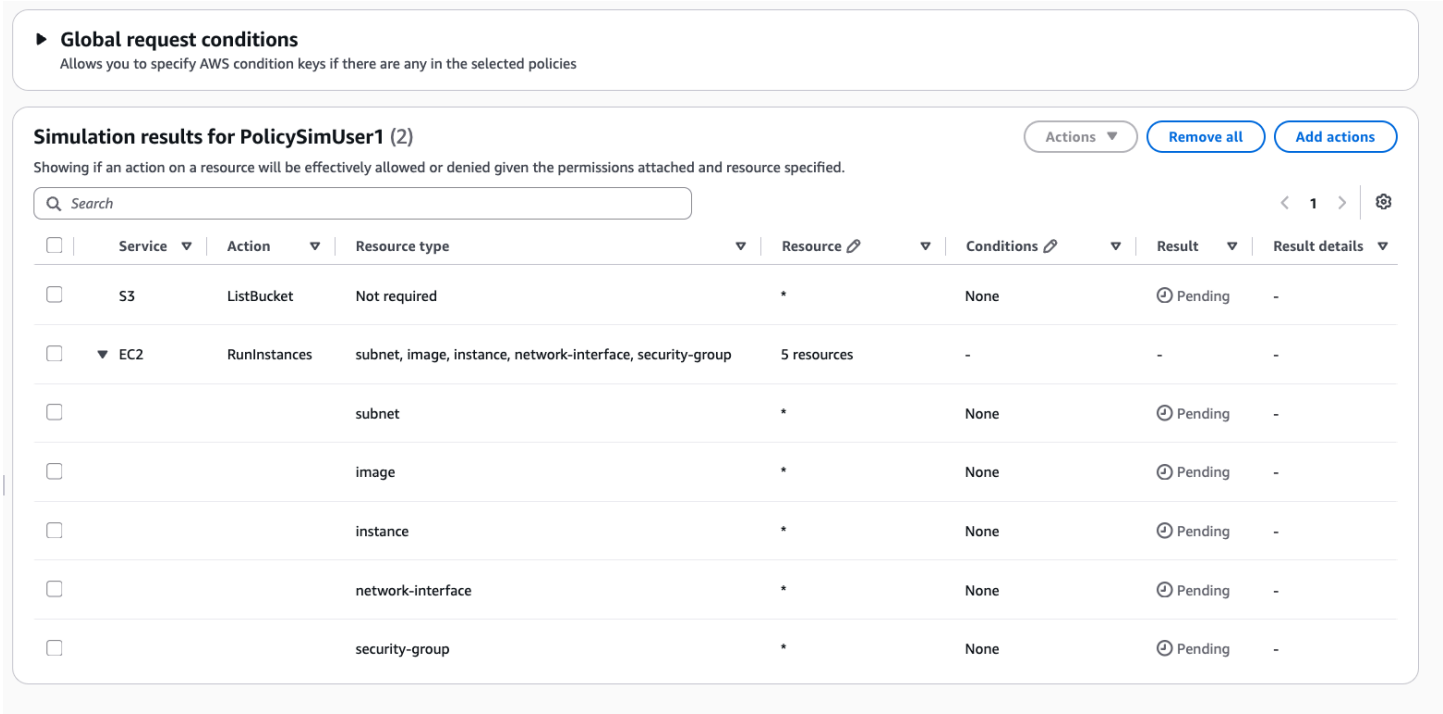


Setelah Anda memilih identitas dan kebijakan di panel kiri, tambahkan tindakan yang ingin Anda uji. Untuk tindakan yang mendukung pemilihan sumber daya, Anda dapat menguji terhadap semua

sumber daya atau memasukkan satu atau lebih ARN sumber daya tertentu. Ketika beberapa tindakan menggunakan jenis sumber daya yang sama, Anda dapat menerapkan satu sumber daya ke semuanya sekaligus, atau mencakupnya ke satu tindakan.



Beberapa tindakan memerlukan sekelompok jenis sumber daya terkait sebelum simulator dapat mengevaluasinya. Misalnya, saat Anda mensimulasikan`ec2:RunInstances`, konsol meminta Anda untuk menentukan gambar, instance, dan grup keamanan bersama-sama, dan subnet atau volume tergantung pada skenario.



Berikan nilai kunci kondisi

Jika kebijakan yang Anda uji menyertakan kunci kondisi, simulator mencantumkan kunci yang relevan dan memungkinkan Anda menetapkan nilai sebelum menjalankan simulasi. Tombol yang

ditampilkan mencerminkan kebijakan yang saat ini dipilih, jadi memilih atau menghapus kebijakan menambah atau menghapus kunci tersebut. Jika suatu nilai tidak memenuhi suatu kondisi, tindakan tersebut ditolak.

Simulator menyediakan nilai untuk kunci kondisi di dua tempat. Kondisi permintaan global berlaku untuk seluruh permintaan simulasi. Ini adalah kunci yang dimulai dengan`aws:`, seperti `aws:MultiFactorAuthPresent` atau`aws:PrincipalServiceNamesList`, dan Anda menetapkan nilainya sekali di area kondisi global. Memilih atau menghapus kebijakan menambah atau menghapus kunci globalnya, dan nilai yang Anda masukkan dipertahankan saat Anda menyesuaikan kebijakan mana yang berada dalam cakupan. Service-specific Kondisi berlaku untuk satu tindakan. Setiap baris tindakan menunjukkan jumlah kondisi yang dirujukkannya, dan Anda membuka baris untuk menyetel kunci khusus layanan untuk tindakan tersebut, seperti `cloudwatch:namespace` untuk `cloudwatch:PutMetricData` atau `ram:PermissionResourceType` untuk. `ram:ListPermissionAssociations` Baris tindakan dapat menyertakan kunci global dan kunci khusus layanan, dan jika ada nilai yang tidak memenuhi suatu kondisi, tindakan tersebut ditolak.

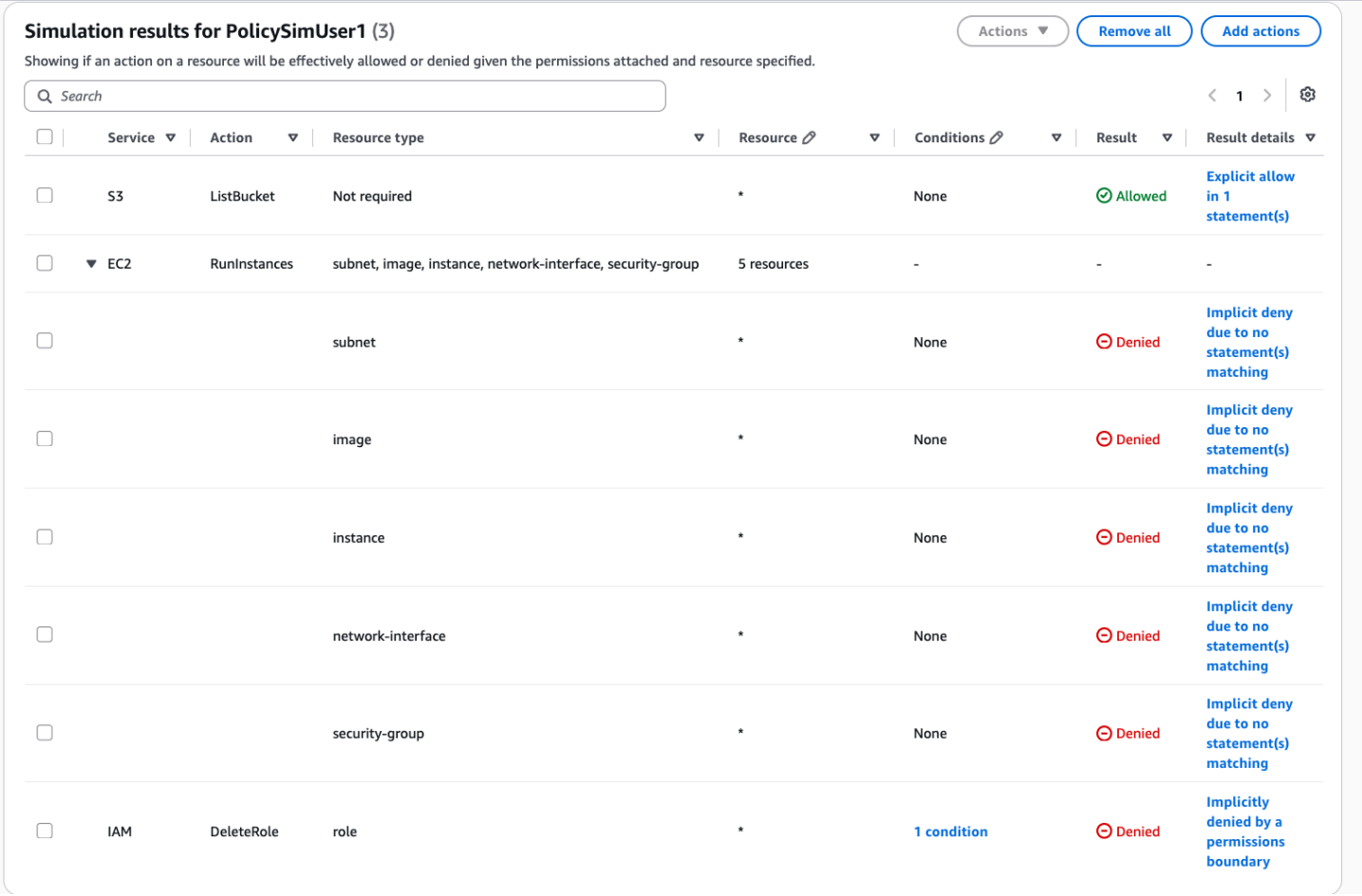
Simulator kebijakan IAM mengisi beberapa kunci konteks utama dan organisasi untuk Anda, dan Anda memberikan nilai untuk sisanya. Untuk daftar kunci yang diisi secara otomatis dan lebih lanjut tentang bagaimana kunci kondisi dievaluasi, lihatCara kerja evaluasi kunci kondisi di simulator kebijakan IAM.

Untuk mempelajari cara menggunakan kunci kondisi, lihatAWS kunci konteks kondisi global.

Menguji batas izin

Ketika identitas memiliki batas izin, itu muncul di panel samping dan disertakan secara default. Anda juga dapat mengecualikan batas dan menjalankan simulasi lagi, jika Anda ingin mengetahui apakah batas adalah alasan tindakan ditolak.

Misalnya, Anda mensimulasikan `iam:DeleteRole` identitas yang batasnya tidak mengizinkannya. Hasilnya adalah penolakan dikaitkan dengan batas izin. Jika Anda kemudian menambahkan kebijakan yang mengizinkan tindakan dan mengecualikan batas, hasilnya berubah menjadi izin. Ini menegaskan bahwa batas adalah satu-satunya kontrol yang menghalangi tindakan.



Sertakan kebijakan kontrol layanan

Jika akun Anda adalah anggota organisasi, Anda dapat menyertakan kebijakan kontrol layanan (SCP) dalam simulasi dari panel samping. Simulator mengevaluasi kunci kondisi yang diisi otomatis dan cakupan sumber daya di SCP saat menentukan hasilnya. Ketika SCP secara eksplisit menyangkal suatu tindakan, hasilnya adalah penolakan eksplisit yang dikaitkan dengan AWS Organizations, dan izin dalam kebijakan lain tidak dapat mengesampingkannya. Matikan kontrol dan jalankan simulasi lagi untuk melihat hasilnya tanpa kontrol organisasi. Untuk daftar kunci kondisi yang diisi secara otomatis, lihatCara kerja evaluasi kunci kondisi di simulator kebijakan IAM.

> **Note**
>
> Untuk alasan keamanan, simulator tidak menampilkan rincian evaluasi SCP. Kunci kondisi yang direferensikan hanya oleh SCP tidak terdaftar bagi Anda untuk menetapkan nilai, dan pernyataan yang cocok yang dikembalikan untuk hasil tidak menyertakan pernyataan

dari SCP. Kunci dan pernyataan yang cocok berasal dari kebijakan berbasis identitas yang melekat pada identitas, dan kebijakan apa pun yang Anda berikan.

Menguji kebijakan berbasis sumber daya

Untuk menyertakan kebijakan berbasis sumber daya dalam simulasi konsol, masukkan ARN sumber daya tertentu dan pilih opsi untuk menyertakan kebijakan sumber daya. Kebijakan berbasis sumber daya muncul di panel samping dan dievaluasi dalam hasilnya. Di konsol, ini tersedia untuk bucket Amazon S3, antrian Amazon SQS, topik Amazon SNS, dan brankas Amazon Glacier.

> **Note**
>
> Simulasi kebijakan berbasis sumber daya tidak didukung untuk peran IAM.

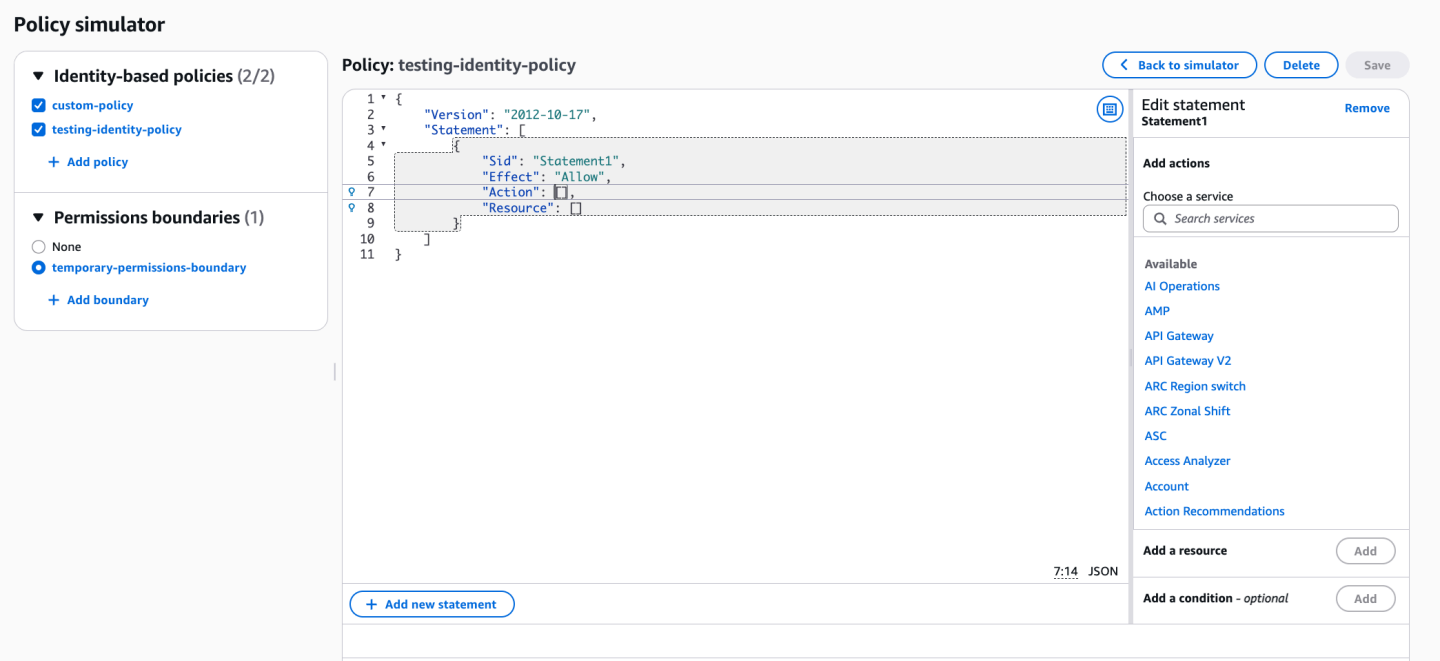
Baca hasil

Setiap baris dalam tabel hasil menunjukkan apakah tindakan diizinkan, ditolak secara implisit, atau ditolak secara eksplisit, bersama dengan penjelasan singkat tentang hasilnya. Gunakan penjelasan untuk mengidentifikasi kebijakan atau kontrol mana yang menentukan hasilnya, seperti izin eksplisit, tidak adanya pernyataan yang cocok, batas izin, atau kontrol organisasi.

Modus kustom

Anda dapat menggunakan editor kebijakan IAM untuk menulis atau menempelkan kebijakan berbasis identitas dan batas izin dalam mode kustom. Kebijakan ini tidak melekat pada identitas dan dapat dievaluasi sebelum Anda menerapkannya. Anda dapat menyertakan beberapa kebijakan berbasis identitas, tetapi hanya satu batas izin untuk simulasi.

> **Note**
>
> Dalam mode Kustom, konsol tidak mendukung penerusan SCP kustom atau mensimulasikan kebijakan berbasis sumber daya. Untuk mensimulasikan SCP kustom atau kebijakan berbasis sumber daya, gunakan API or. AWS CLI AWS



Simulasikan kebijakan (AWS CLI and AWS API)

Mensimulasikan kebijakan dengan AWS API AWS CLI atau umumnya membutuhkan dua langkah:

1. Evaluasi kebijakan dan kembalikan daftar kunci konteks yang mereka referensikan, sehingga Anda tahu nilai kunci konteks mana yang akan disediakan. Langkah ini bersifat opsional.
2. Simulasikan kebijakan, menyediakan tindakan, sumber daya, dan nilai kunci konteks untuk digunakan selama simulasi.

Setiap tindakan dipasangkan dengan setiap sumber daya, dan simulasi menentukan apakah kebijakan mengizinkan atau menolak tindakan tersebut untuk sumber daya tersebut. Anda dapat memberikan nilai untuk kunci konteks apa pun yang dirujuk kebijakan Anda. Jika Anda tidak memberikan nilai untuk kunci konteks, simulasi masih berjalan dan mengembalikan penolakan implisit, dan kunci yang hilang dikembalikan `missingContextValues` sehingga Anda tahu nilai mana yang akan ditambahkan. Jika kunci konteks yang hilang direferensikan hanya oleh SCP, kunci itu tidak dikembalikan karena alasan keamanan, tetapi Anda masih menerima keputusan.

Tindakan API dibagi menjadi dua kelompok:

- Tindakan yang mensimulasikan hanya kebijakan yang diteruskan langsung ke API sebagai string: GetContextKeysForCustomPolicy dan SimulateCustomPolicy.
- Tindakan yang mensimulasikan kebijakan yang dilampirkan ke pengguna IAM tertentu, grup pengguna, peran, atau sumber daya: GetContextKeysForPrincipalPolicy dan

SimulatePrincipalPolicy. Karena ini dapat mengungkapkan izin yang ditetapkan ke entitas IAM lainnya, pertimbangkan untuk membatasi akses ke entitas tersebut.

Tindakan ini juga mendukung hal-hal berikut:

- `OrderedOrganizationPolicyInputListon` SimulateCustomPolicy memungkinkan Anda mensimulasikan efek hierarki SCP organisasi tanpa menggunakan prinsipal langsung.
- `PolicyExclusionListon` SimulatePrincipalPolicy memungkinkan Anda menguji "bagaimana jika saya menghapus kebijakan ini?" skenario tanpa mengubah kebijakan mana yang melekat pada prinsipal.
- Respons mengembalikan satu `EvaluationResult` per tindakan, dengan keputusan agregat di tingkat atas dan detail per sumber daya di `ResourceSpecificResults` dalamnya.

AWS CLI and AWS Referensi API

Untuk contoh AWS CLI perintah untuk mensimulasikan kebijakan, lihat berikut ini:

- AWS CLI: dapatkan-konteks- kunci-untuk-kebijakan, dapatkan-kunci-konteks-untuk-kebijakan utama, simulasi-kebijakan-kustom, simulasi-prinsip-kebijakan https://docs.aws.amazon.com/cli/latest/reference/iam/get-context-keys-for-principal-policy.html https://docs.aws.amazon.com/cli/latest/reference/iam/simulate-custom-policy.html https://docs.aws.amazon.com/cli/latest/reference/iam/simulate-principal-policy.html
- AWS API: GetContextKeysForCustomPolicy, GetContextKeysForPrincipalPolicy, SimulateCustomPolicy, SimulatePrincipalPolicy

Menambahkan dan menghapus izin identitas IAM

Anda menggunakan kebijakan untuk menentukan izin bagi identitas (pengguna, grup pengguna, atau peran). Anda dapat menambah dan menghapus izin dengan melampirkan dan melepaskan kebijakan IAM untuk identitas menggunakan, AWS Command Line Interface (AWS CLI) Konsol Manajemen AWS, atau API. AWS Anda juga dapat menggunakan kebijakan untuk menetapkan batas izin hanya untuk entitas (pengguna atau peran) yang menggunakan metode yang sama. Batas izin adalah AWS fitur lanjutan yang mengontrol izin maksimum yang dapat dimiliki entitas.

Topik

- Terminologi

- Lihat aktivitas identitas
- Menambahkan izin identitas IAM (konsol)
- Menghapus izin identitas IAM (konsol)
- Menambahkan kebijakan IAM ()AWS CLI
- Menghapus kebijakan IAM ()AWS CLI
- Menambahkan kebijakan IAM (AWS API)
- Menghapus kebijakan IAM (AWS API)

Terminologi

Saat Anda mengaitkan kebijakan izin dengan identitas (pengguna IAM, grup IAM, dan peran IAM), terminologi dan prosedur bervariasi tergantung pada apakah Anda bekerja dengan kebijakan terkelola atau sebaris:

- Pasang – Dipakai dengan kebijakan terkelola. Anda melampirkan kebijakan terkelola ke identitas (pengguna, grup pengguna, atau peran). Melampirkan kebijakan menerapkan izin dari kebijakan ke identitas.
- Lepas – Dipakai dengan kebijakan terkelola. Anda melepaskan kebijakan terkelola dari identitas IAM (pengguna, grup pengguna, atau peran). Melepas kebijakan menghapus izinnya dari identitas.
- Ditanamkan – Dipakai dengan kebijakan yang selaras. Anda menyematkan kebijakan inline pada sebuah identitas (pengguna, grup pengguna, atau peran). Menanamkan kebijakan berarti menerapkan izin kebijakan ke identitas. Karena kebijakan inline disimpan di identitas, ia disematkan dan bukan ditempelkan, walau hasilnya serupa.

  > **Note**
  >
  > Anda bisa menanamkan kebijakan inline peran terkait layanan hanya dalam layanan yang bergantung pada peran tersebut. Lihat dokumentasi AWS dari layanan Anda untuk melihat apakah ia mendukung fitur ini.

- Hapus – Dipakai dengan kebijakan yang selaras. Anda menghapus kebijakan sebaris dari identitas IAM (pengguna, grup pengguna, atau peran). Menghapus kebijakan akan menghapus izinnya dari identitas.

> **Note**
>
> Anda bisa menghapus kebijakan inline untukperan terkait layanan hanya dalam layanan yang bergantung pada peran tersebut. Lihat dokumentasi AWS dari layanan Anda untuk melihat apakah ia mendukung fitur ini.

Anda dapat menggunakan konsol, AWS CLI, atau AWS API untuk melakukan salah satu tindakan ini.

Informasi selengkapnya

- Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan inline, lihat Kebijakan terkelola dan kebijakan inline.
- Untuk informasi lebih lanjut tentang batasan izin, lihat Batas izin untuk entitas IAM.
- Untuk informasi umum tentang kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management.
- Untuk informasi tentang memvalidasi kebijakan IAM, lihat. Validasi kebijakan IAM
- Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Lihat aktivitas identitas

Sebelum mengubah izin untuk identitas (pengguna, grup pengguna, atau peran), Anda harus melihat aktivitas tingkat layanan mereka yang terkini. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya tentang melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menambahkan izin identitas IAM (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk menambahkan izin ke identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, tempelkan kebijakan terkelola yang mengendalikan izin, atau tentukan kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk identitas (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih tombol radio di sebelah nama kebijakan yang akan dilampirkan. Anda dapat menggunakan kotak pencarian untuk menyaring daftar kebijakan.
4. Pilih Tindakan, lalu pilih Lampirkan.
5. Pilih satu atau lebih identitas untuk melampirkan kebijakan. Anda bisa memakai kotak pencarian untuk memfilter daftar entitas penanggung jawab. Setelah memilih identitas, pilih Lampirkan kebijakan.

Untuk menggunakan kebijakan terkelola untuk menetapkan batasan izin (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman detail kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin dan pilih Tetapkan kebijakan ini sebagai batas izin.
5. Pilih satu atau lebih pengguna atau peran untuk menggunakan kebijakan sebagai batasan izin. Anda bisa memakai kotak pencarian untuk memfilter daftar entitas penanggung jawab. Setelah memilih prinsipal, pilih Setel batas izin.

Untuk menyematkan kebijakan inline bagi pengguna atau peran (konsole)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna atau Peran.
3. Dari daftar, pilih nama pengguna atau peran untuk menanamkan kebijakan ke dalamnya.
4. Pilih tab Izin.
5. Pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.

> **Note**
>
> Anda tidak menanam kebijakan yang selaras padaperan terkait-layanan di IAM. Karena layanan terkait menentukan apakah Anda bisa mengubah izin peran tersebut, Anda mungkin bisa menambah kebijakan tambahan dari konsol layanan, API, atau AWS CLI. Untuk melihat dokumentasi peran terkait-layanan untuk layanan, lihat AWS layanan yang bekerja dengan IAM dan pilih Ya pada kolom Peran Terkait Layanan untuk layanan Anda.

6. Pilih dari metode berikut untuk melihat langkah-langkah yang diperlukan untuk membuat kebijakan Anda:

   - Mengimpor kebijakan terkelola yang ada – Anda bisa mengimpor kebijakan yang dikelola di dalam akun Anda dan lalu mengedit kebijakan itu untuk menyesuaikan dengan kebutuhan khusus Anda. Kebijakan terkelola dapat berupa kebijakan AWS terkelola atau kebijakan terkelola pelanggan yang Anda buat sebelumnya.
   - Membuat kebijakan dengan editor visual – Anda bisa membuat kebijakan baru dari nol di editor visual. Jika Anda memakai editor visual, Anda tidak harus memahami sintaksis JSON.
   - Membuat kebijakan menggunakan editor JSON— Dalam opsi editor JSON, Anda dapat menggunakan sintaks JSON untuk membuat kebijakan. Anda bisa mengetik dokumen kebijakan JSON yang baru atau menempelkan satucontoh kebijakan.

7. Setelah Anda membuat kebijakan inline, ia akan secara otomatis tertanam di pengguna atau peran Anda.

Untuk menanam kebijakan inline bagi grup pengguna (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih User groups (Grup pengguna).
3. Dari daftar, pilih nama pengguna dari grup pengguna untuk menyematkan kebijakan ke dalamnya.
4. Pilih tab Permissions (Izin), pilih Add permissions (Tambahkan izin), lalu pilih Create inline policy (Membuat kebijakan inline).
5. Lakukan salah satu tindakan berikut:

- Pilih opsi Visual untuk membuat kebijakan. Untuk informasi selengkapnya, lihat Membuat kebijakan dengan editor visual.
- Pilih opsi JSON untuk membuat kebijakan. Untuk informasi selengkapnya, lihat Membuat kebijakan menggunakan editor JSON.

6. Jika Anda puas dengan kebijakan ini, pilih Create policy (Buat kebijakan).

Untuk mengubah batasan izin untuk satu atau lebih entitas (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman detail kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin. Pilih kotak centang di sebelah pengguna atau peran yang batasnya ingin Anda ubah, lalu pilih Ubah.
5. Pilih kebijakan yang akan dipakai sebagai batasan izin. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan. Setelah memilih kebijakan, pilih Setel batas izin.

Menghapus izin identitas IAM (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk menghapus izin dari identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, lepaskan kebijakan terkelola yang mengendalikan izin, atau hapus kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menghapus kebijakan inline.

Untuk melepas kebijakan terkelola yang digunakan sebagai kebijakan perizinan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih tombol radio di sebelah nama kebijakan yang akan dilepas. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Actions (Tindakan), lalu pilih Detach (Lepas).

5. Pilih identitas yang ingin dilepas kebijakannya. Anda bisa memakai kotak pencarian untuk memfilter daftar identitas. Setelah memilih identitas, pilih Lepaskan kebijakan.

Untuk menghapus batasan perizinan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan yang ingin diatur. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pada halaman ringkasan kebijakan, pilih tab Entitas yang dilampirkan, lalu, jika perlu, buka bagian Terlampir sebagai batas izin dan pilih entitas untuk menghapus batas izin. Kemudian pilih Hapus batas.
5. Konfirmasikan bahwa Anda ingin menghapus batas dan memilih Hapus batas.

Untuk menghapus kebijakan inline (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, atauPeran.
3. Dari daftar, pilih nama grup pengguna, pengguna, atau peran yang memiliki kebijakan yang ingin Anda hapus.
4. Pilih tab Izin.
5. Pilih kotak centang di samping kebijakan dan pilih Hapus.
6. Pilih Hapus di kotak konfirmasi.

Menambahkan kebijakan IAM ()AWS CLI

Anda dapat menggunakan AWS CLI untuk menambahkan izin ke identitas (pengguna, grup pengguna, atau peran). Untuk melakukan hal ini, tempelkan kebijakan terkelola yang mengendalikan izin, atau tentukan kebijakan yang berfungsi sebagai batasan perizinan. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk entitas (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: ambil-kebijakan
2. Untuk melampirkan kebijakan terkelola ke identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
   - aws iam attach-user-policy
   - aws iam attach-group-policy
   - aws iam attach-role-policy

Untuk menggunakan kebijakan terkelola untuk menetapkan batasan perizinan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan
2. Untuk memakai kebijakan yang dikelola pada pengaturan batasan izin untuk entitas (pengguna atau peran), gunakan salah satu perintah berikut:
   - aws iam put-user-permissions-boundary
   - aws iam put-role-permissions-boundary

Untuk menyematkan kebijakan inline (AWS CLI)

Untuk menanamkan kebijakan inline untuk identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:

- aws iam put-user-policy
- aws iam put-group-policy
- aws iam put-role-policy

Menghapus kebijakan IAM ()AWS CLI

Anda dapat menggunakan AWS CLI untuk melepaskan kebijakan terkelola yang mengontrol izin, atau menghapus kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menghapus kebijakan inline.

Untuk melepas kebijakan terkelola yang digunakan sebagai kebijakan perizinan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan
2. (Opsional) Untuk mengetahui hubungan antara kebijakan dan identitas, jalankan perintah berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan terkelola:
     - aws iam list-entities-for-policy
   - Untuk mendaftar kebijakan terkelola yang menempel pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
     - aws iam list-attached-user-policies
     - aws iam list-attached-group-policies
     - aws iam list-attached-role-policies
3. Untuk melepas kebijakan terkelola dari identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:
   - aws iam detach-user-policy
   - aws iam detach-group-policy
   - aws iam detach-role-policy

Untuk menghapus batasan perizinan (AWS CLI)

1. (Opsional) Untuk melihat kebijakan terkelola mana yang saat ini dipakai untuk mengatur batasan izin bagi pengguna atau peran, jalankan perintah berikut:
   - aws iam dapatkan-pengguna
   - aws iam mendapatkan peran

2. (Opsional) Untuk melihat pengguna atau peran dimana kebijakan terkelola dipakai untuk batasan izin, jalankan perintah berikut:

   - aws iam list-entities-for-policy

3. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, jalankan perintah berikut:

   - Untuk mencantumkan kebijakan terkelola: aws iam daftar-kebijakan
   - Untuk mengambil informasi rinci tentang kebijakan yang dikelola: aws-iam ambil-kebijakan

4. Untuk menghapus batasan izin dari pengguna atau peran, gunakan salah satu perintah berikut:

   - aws iam delete-user-permissions-boundary
   - aws iam delete-role-permissions-boundary

Untuk menghapus kebijakan inline (AWS CLI)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies

2. (Opsional) Untuk mengambil dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. Untuk menghapus kebijakan inline dari identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:

   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Menambahkan kebijakan IAM (AWS API)

Anda dapat menggunakan AWS API untuk melampirkan kebijakan terkelola yang mengontrol izin atau menentukan kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menyematkan kebijakan inline.

Untuk menggunakan kebijakan terkelola sebagai kebijakan izin untuk entitas (AWS API)

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh kembali informasi detail tentang kebijakan yang dikelola: GetPolicy
2. Untuk melampirkan kebijakan terkelola pada sebuah identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:
   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Untuk menggunakan kebijakan terkelola untuk menetapkan batas izin (API)AWS

1. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh kembali informasi detail tentang kebijakan yang dikelola: GetPolicy
2. Untuk menggunakan kebijakan terkelola pada penetapan batas perizinan bagi suatu entitas (pengguna atau peran), hubungi salah satu layanan berikut:
   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Untuk menyematkan kebijakan inline (API)AWS

Untuk menyimpan kebijakan inline dalam suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:

- PutUserPolicy
- PutGroupPolicy

- PutRolePolicy

Menghapus kebijakan IAM (AWS API)

Anda dapat menggunakan AWS API untuk melepaskan kebijakan terkelola yang mengontrol izin atau menghapus kebijakan yang berfungsi sebagai batas izin. Anda juga bisa menghapus kebijakan inline.

Untuk melepaskan kebijakan terkelola yang digunakan sebagai kebijakan izin (API)AWS

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh kembali informasi detail tentang kebijakan yang dikelola: GetPolicy
2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan terkelola:
     - ListEntitiesForPolicy
   - Untuk mencantumkan kebijakan terkelola pada suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Untuk melepas kebijakan terkelola dari suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:
   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

Untuk menghapus batas izin (API)AWS

1. Untuk melihat kebijakan terkelola yang sedang digunakan pada penetapan batas perizinan bagi seorang pengguna atau peran, hubungi salah satu layanan berikut:

- GetUser
- GetRole

2. (Opsional) Untuk melihat pengguna atau peran yang menggunakan kebijakan terkelola pada batas perizinan, hubungi layanan berikut:

   - ListEntitiesForPolicy

3. (Opsional) Untuk melihat informasi tentang kebijakan yang dikelola, hubungi layanan berikut:

   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh informasi detail tentang kebijakan yang dikelola: GetPolicy

4. Untuk menghilangkan batas perizinan dari pengguna atau peran, hubungi salah satu layanan berikut:

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Untuk menghapus kebijakan inline (AWS API)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (Opsional) Untuk memperoleh dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

3. Untuk menghapus kebijakan inline dari suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:

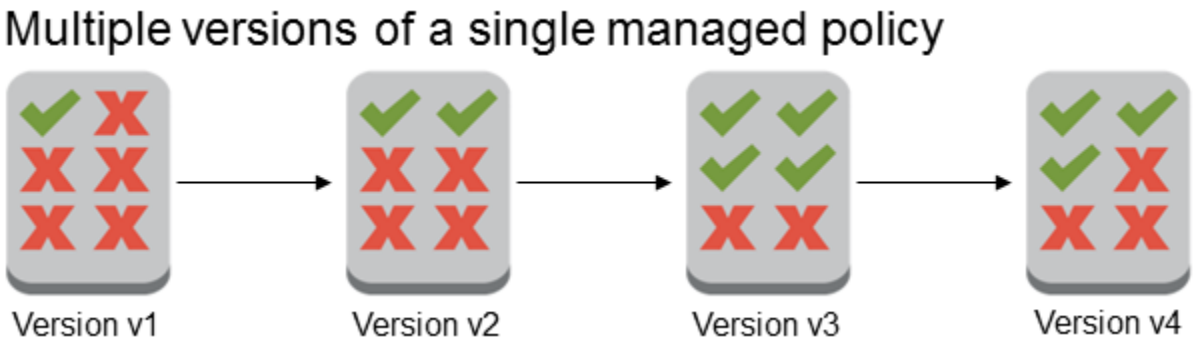
   - DeleteUserPolicy
   - DeleteGroupPolicy

- DeleteRolePolicy

Peningkatan versi IAM

Saat Anda membuat perubahan pada kebijakan terkelola pelanggan IAM, dan saat AWS membuat perubahan pada kebijakan AWS terkelola, kebijakan yang diubah tidak akan menimpa kebijakan yang ada. Sebaliknya, IAM membuat versi baru dari kebijakan terkelola. IAM menyimpan sampai dengan lima versi kebijakan terkelola pelanggan Anda. IAM tidak mendukung peningkatan versi untuk kebijakan selaras.

Diagram berikut mengilustrasikan versioning untuk kebijakan terkelola pelanggan. Dalam contoh ini, versi 1-4 disimpan. Anda dapat menyimpan hingga lima versi kebijakan terkelola ke IAM. Saat mengedit kebijakan yang akan membuat versi tersimpan keenam, Anda dapat memilih versi lama yang tidak akan disimpan lagi. Anda dapat kembali ke salah satu dari empat versi tersimpan lainnya kapan saja.



Versi kebijakan berbeda dari elemen kebijakan `Version`. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Untuk mempelajari lebih lanjut tentang `Version` elemen kebijakan lihat Elemen kebijakan IAM JSON: Version.

Anda dapat menggunakan beberapa versi untuk melacak perubahan pada kebijakan terkelola. Misalnya, Anda mungkin melakukan perubahan pada kebijakan terkelola dan kemudian menemukan bahwa perubahan tersebut memiliki efek yang tak diinginkan. Dalam kejadian ini, Anda dapat kembali ke kebijakan terkelola versi sebelumnya dengan mengatur versi sebelumnya sebagai versi default.

Topik berikut menjelaskan bagaimana Anda dapat menggunakan pembuatan versi untuk kebijakan terkelola.

Topik

- Pembatasan versi

- Gunakan versi untuk memutar kembali perubahan
- Izin untuk mengatur versi default kebijakan
- Mengatur versi default kebijakan terkelola pelanggan

Pembatasan versi

Kebijakan terkelola dapat memiliki sampai dengan lima versi. Jika Anda perlu membuat perubahan pada kebijakan terkelola di luar lima versi dari AWS Command Line Interface, atau AWS API, Anda harus menghapus satu atau beberapa versi yang ada terlebih dahulu. Jika Anda menggunakan Konsol Manajemen AWS, Anda tidak perlu menghapus versi sebelum mengedit kebijakan Anda. Ketika Anda menyimpan versi keenam, muncul kotak dialog yang menyarankan Anda menghapus satu atau beberapa versi non-default kebijakan Anda. Anda dapat melihat berkas kebijakan JSON untuk setiap versi untuk membantu Anda mengambil keputusan. Untuk detail tentang kotak dialog ini, lihat the section called “Edit kebijakan IAM”.

Anda dapat menghapus versi kebijakan terkelola apa pun yang Anda inginkan, kecuali versi default. Ketika Anda menghapus suatu versi, pengidentifikasi versi untuk sisanya tidak berubah. Hasilnya, pengidentifikasi versi mungkin tak berurutan. Sebagai contoh, apabila Anda menghapus kebijakan terkelola versi v2 dan v4 dan menambahkan dua versi baru, pengidentifikasi versi lainnya mungkin adalah v1, v3, v5, v6, dan v7.

Gunakan versi untuk memutar kembali perubahan

Anda dapat mengatur versi default kebijakan terkelola pelanggan untuk mengembalikan perubahan Anda. Sebagai contoh, pertimbangkan alur perencanaan berikut ini:

Anda membuat kebijakan terkelola pelanggan yang memungkinkan pengguna mengelola bucket Amazon S3 tertentu menggunakan. Konsol Manajemen AWS Setelah pembuatan, kebijakan terkelola pelanggan Anda hanya memiliki satu versi, yang diidentifikasi sebagai v1, sehingga versi tersebut secara otomatis diatur sebagai default. Kebijakan ini bekerja sebagaimana mestinya.

Kemudian, Anda memperbaharui kebijakan untuk menambahkan izin mengelola bucket Amazon S3 kedua. IAM membuat versi baru kebijakan, yang diidentifikasi sebagai v2, berisikan perubahan Anda. Anda mengatur versi v2 sebagai standar, dan tidak lama kemudian pengguna Anda melaporkan bahwa mereka tidak memiliki izin untuk menggunakan konsol Amazon S3. Dalam hal ini, Anda dapat kembali ke kebijakan versi v1, yang Anda tahu bekerja sebagaimana mestinya. Untuk melakukannya, Anda menetapkan versi v1 sebagai versi standar. Pengguna Anda sekarang dapat menggunakan konsol Amazon S3 untuk mengelola bucket asli.

Kemudian, setelah Anda menentukan kesalahan dalam kebijakan versi v2, Anda memperbaharui kebijakannya lagi untuk menambahkan izin mengelola bucket Amazon S3 yang kedua. IAM membuat versi baru kebijakan lainnya, yang diidentifikasi sebagai v3. Anda mengatur versi v3 sebagai standar, dan versi ini berjalan sebagaimana mestinya. Pada tahap ini, Anda menghapus versi v2 kebijakan.

Izin untuk mengatur versi default kebijakan

Izin yang diperlukan untuk mengatur versi default kebijakan sesuai dengan operasi API AWS untuk tugas tersebut. Anda dapat menggunakan operasi API `CreatePolicyVersion` atau `SetDefaultPolicyVersion` untuk mengatur versi default kebijakan. Untuk memperbolehkan seseorang mengatur versi kebijakan default dari kebijakan yang ada, Anda dapat mengizinkan akses untuk tindakan `iam:CreatePolicyVersion` atau tindakan `iam:SetDefaultPolicyVersion`. Tindakan `iam:CreatePolicyVersion` memungkinkan mereka membuat kebijakan versi baru dan mengatur versi tersebut sebagai default. Tindakan `iam:SetDefaultPolicyVersion` memungkinkan mereka mengatur versi kebijakan yang ada sebagai kebijakan default.

> **Important**
>
> Untuk mencegah pengguna membuat perubahan pada versi default kebijakan, Anda harus menolak keduanya `iam:CreatePolicyVersion` dan`iam:SetDefaultPolicyVersion`.

Anda dapat menggunakan kebijakan berikut untuk menolak akses pengguna untuk mengubah kebijakan terkelola pelanggan yang sudah ada:

JSON

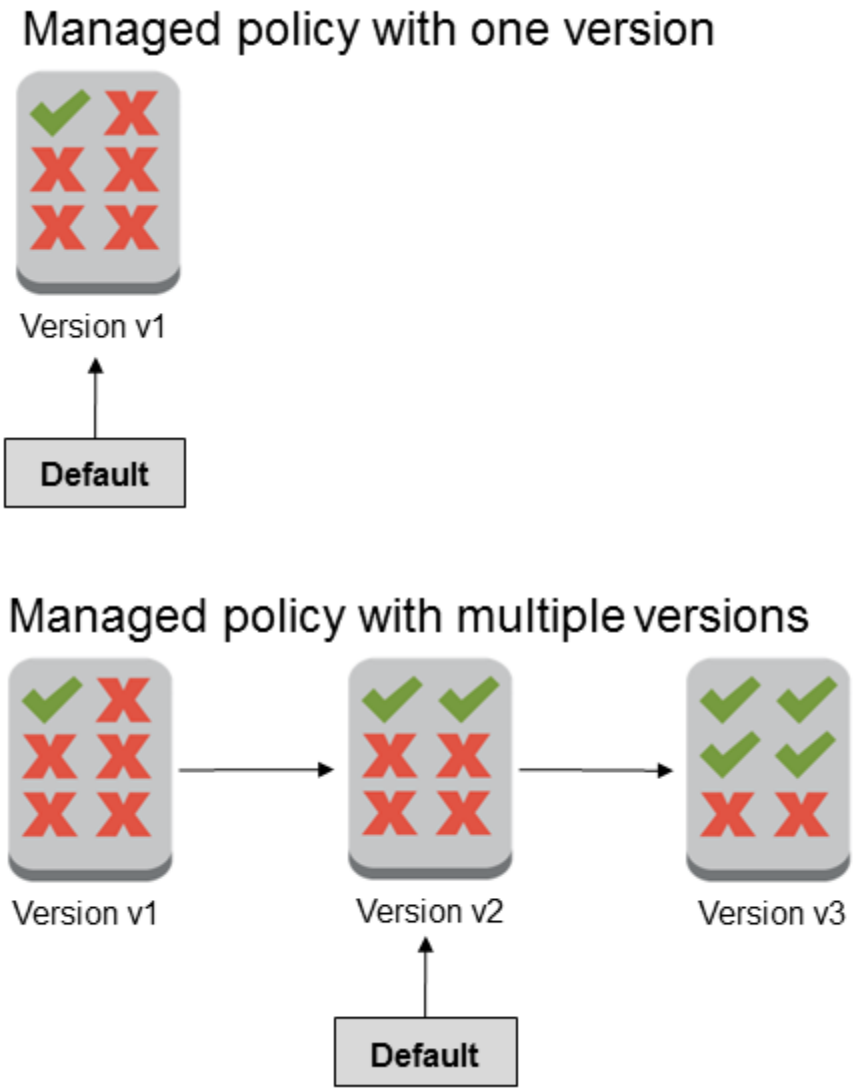
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
```

```
}
```

Mengatur versi default kebijakan terkelola pelanggan

Salah satu versi kebijakan terkelola ditetapkan sebagai versi default. Versi default kebijakan adalah versi operatif—yaitu, ini adalah versi yang berlaku untuk semua entitas utama (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan pada kebijakan terkelola.

Ketika Anda membuat kebijakan terkelola pelanggan, kebijakan tersebut dimulai versi tunggal yang diidentifikasi sebagai v1. Untuk kebijakan terkelola dengan versi tunggal saja, versi tersebut secara otomatis diatur sebagai default. Untuk kebijakan terkelola pelanggan dengan lebih dari satu versi, Anda memilih versi mana yang akan ditetapkan sebagai default. Untuk kebijakan AWS terkelola, versi default ditetapkan oleh AWS. Diagram berikut mengilustrasikan hal ini.



Anda dapat menyetel versi default kebijakan terkelola pelanggan untuk menerapkan versi tersebut ke setiap identitas IAM (pengguna, grup pengguna, dan peran) tempat kebijakan dilampirkan. Anda tidak dapat menyetel versi default untuk kebijakan AWS terkelola atau kebijakan sebaris.

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk diatur menjadi versi default. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Versi kebijakan. Pilih kotak centang di sebelah versi yang ingin Anda tetapkan sebagai versi default, lalu pilih Set as default.

Untuk mempelajari cara menyetel versi default kebijakan terkelola pelanggan dari AWS Command Line Interface atau AWS API, lihatEdit kebijakan IAM ()AWS CLI.

Edit kebijakan IAM

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Kebijakan disimpan AWS sebagai dokumen JSON dan dilampirkan pada prinsipal sebagai kebijakan berbasis identitas di IAM. Anda dapat melampirkan kebijakan berbasis identitas ke prinsipal (atau identitas), seperti grup pengguna IAM, pengguna, atau peran. Kebijakan berbasis identitas mencakup kebijakan terkelola, kebijakan yang AWS dikelola pelanggan, dan kebijakan inline. Anda dapat mengedit kebijakan terkelola pelanggan dan kebijakan inline di IAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Umumnya lebih baik menggunakan kebijakan yang dikelola pelanggan daripada kebijakan inline atau kebijakan AWS terkelola. AWS kebijakan terkelola biasanya memberikan izin administratif atau hanya-baca yang luas. Kebijakan inline tidak dapat digunakan kembali pada identitas lain atau dikelola di luar identitas di mana mereka ada. Untuk keamanan terbesar, berikan hak istimewa paling sedikit, yang berarti hanya memberikan izin yang diperlukan untuk melakukan tugas pekerjaan tertentu.

Ketika Anda membuat atau mengedit kebijakan IAM, AWS dapat secara otomatis melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif dengan hak istimewa paling sedikit dalam pikiran. Dalam Konsol Manajemen AWS, IAM mengidentifikasi kesalahan

sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan Anda lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Anda dapat menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline di IAM. Untuk informasi selengkapnya tentang menggunakan CloudFormation templat untuk menambah atau memperbarui kebijakan, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan CloudFormation Pengguna.

Topik

- Edit kebijakan IAM (konsol)
- Edit kebijakan IAM ()AWS CLI
- Edit kebijakan IAM (AWS API)

Edit kebijakan IAM (konsol)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan Konsol Manajemen AWS untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline di IAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan Referensi elemen kebijakan IAM JSON.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menyunting kebijakan terkelola pelanggan (konsol)

Anda dapat mengedit kebijakan terkelola pelanggan untuk mengubah izin yang ditentukan dalam kebijakan dari Konsol Manajemen AWS. Kebijakan terkelola pelanggan dapat memiliki sampai

dengan lima versi. Hal ini penting karena apabila Anda melakukan perubahan pada kebijakan terkelola lebih dari lima versi, Konsol Manajemen AWS akan menyarankan agar Anda memutuskan versi mana yang akan dihapus. Anda juga dapat mengubah versi default atau menghapus suatu versi kebijakan sebelum Anda menyuntingnya untuk menghindari anjuran. Untuk mempelajari lebih lanjut tentang versi, lihat Peningkatan versi IAM.

Console

Untuk mengedit kebijakan terkelola pelanggan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk disunting. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Izin, lalu pilih Edit.
5. Lakukan salah satu tindakan berikut:
   - Pilih opsi Visual untuk mengubah kebijakan Anda tanpa memahami sintaks JSON. Anda dapat membuat perubahan pada layanan, tindakan, sumber daya, atau kondisi opsional untuk setiap blokir izin dalam kebijakan Anda. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Setelah selesai melakukan perubahan, pilih Berikutnya untuk melanjutkan.
   - Pilih opsi JSON untuk mengubah kebijakan Anda dengan mengetik atau menempelkan teks di kotak teks JSON. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

     > **Note**
     >
     > Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

6. Pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

7. Jika kebijakan terkelola sudah memiliki maksimal lima versi, memilih Simpan perubahan akan menampilkan kotak dialog. Untuk menyimpan versi baru Anda, versi kebijakan non-default tertua akan dihapus dan diganti dengan versi baru ini. Secara opsional, Anda dapat mengatur versi baru sebagai versi kebijakan default.

   Pilih Simpan perubahan untuk menyimpan versi kebijakan baru Anda.

Menyetel versi default kebijakan terkelola pelanggan (konsol)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari Konsol Manajemen AWS. Anda dapat menggunakan kebijakan ini untuk membuat konfigurasi dasar yang konsisten untuk izin di seluruh organisasi Anda. Semua lampiran baru kebijakan akan menggunakan set izin standar ini.

Console

Untuk mengatur versi default kebijakan terkelola pelanggan (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dari daftar kebijakan, pilih nama kebijakan untuk diatur menjadi versi default. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Versi kebijakan. Centang kotak di samping versi yang ingin Anda tetapkan sebagai versi default, lalu pilih Atur sebagai default.

Menghapus versi kebijakan terkelola pelanggan (konsol)

Anda mungkin perlu menghapus versi kebijakan yang dikelola pelanggan untuk menghapus izin yang sudah ketinggalan zaman atau salah yang tidak lagi diperlukan atau menimbulkan potensi risiko keamanan. Dengan hanya mempertahankan versi yang diperlukan, Anda dapat membantu memastikan bahwa Anda tetap berada dalam batas lima versi kebijakan terkelola, sehingga memungkinkan ruang untuk pembaruan dan penyempurnaan di masa mendatang. Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari Konsol Manajemen AWS.

Console

Untuk menghapus versi kebijakan terkelola pelanggan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Pilih nama kebijakan terkelola pelanggan dengan versi yang ingin Anda hapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Versi kebijakan. Centang kotak di samping versi yang ingin Anda hapus. Kemudian pilih Hapus.
5. Konfirmasi bahwa Anda ingin menghapus versi tersebut, lalu pilih Hapus.

Mengedit kebijakan inline (konsol)

Anda mungkin perlu mengedit kebijakan terkelola pelanggan untuk memperbarui atau menyempurnakan izin yang diberikan, memastikan mereka tetap selaras dengan persyaratan keamanan organisasi Anda yang terus berkembang dan kebutuhan kontrol akses. Pengeditan memungkinkan Anda menyesuaikan dokumen JSON kebijakan, menambahkan, memodifikasi, atau menghapus tindakan, sumber daya, atau kondisi tertentu untuk mempertahankan prinsip hak istimewa yang paling rendah dan beradaptasi dengan perubahan di lingkungan atau proses Anda. Anda dapat mengedit kebijakan inline dari Konsol Manajemen AWS.

Console

Untuk mengedit kebijakan inline untuk pengguna, grup pengguna, atau peran

1. Di panel navigasi, pilih Pengguna, Grup Pengguna, atauPeran.
2. Dari daftar, pilih nama pengguna, grup pengguna, atau peran dengan kebijakan yang ingin Anda modifikasi. Lalu pilih tabPerizinan dan kembangkan kebijakan.
3. Untuk mengedit kebijakan inline, pilih Edit Policy (Edit Kebijakan).
4. Lakukan salah satu tindakan berikut:
   - Pilih opsi Visual untuk mengubah kebijakan Anda tanpa memahami sintaks JSON. Anda dapat membuat perubahan pada layanan, tindakan, sumber daya, atau kondisi opsional untuk setiap blokir izin dalam kebijakan Anda. Anda juga dapat mengimpor kebijakan

untuk menambahkan izin tambahan ke bawah kebijakan Anda. Setelah selesai melakukan perubahan, pilih Berikutnya untuk melanjutkan.

- Pilih opsi JSON untuk mengubah kebijakan Anda dengan mengetik atau menempelkan teks di kotak teks JSON. Anda juga dapat mengimpor kebijakan untuk menambahkan izin tambahan ke bawah kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Untuk menyimpan perubahan Anda tanpa memengaruhi entitas yang terlampir sekarang, kosongkan kotak centang untuk Simpan sebagai versi standar.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

5. Pada halaman Tinjauan, tinjau ringkasan kebijakan, lalu pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Edit kebijakan IAM ()AWS CLI

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline di IAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan Referensi elemen kebijakan IAM JSON.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Mengedit kebijakan terkelola pelanggan (AWS CLI)

Anda dapat mengedit kebijakan yang dikelola pelanggan dari AWS CLI.

> **Note**
>
> Kebijakan terkelola dapat memiliki hingga lima versi. Apabila Anda perlu melakukan perubahan pada kebijakan terkelola pelanggan lebih dari lima versi, Anda harus terlebih dahulu menghapus satu atau beberapa versi yang ada.

Untuk mengedit kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: daftar-kebijakan
   - Untuk memperoleh informasi terperinci tentang kebijakan terkelola: dapatkan-kebijakan
2. (Opsional) Untuk mengetahui tentang hubungan antara kebijakan dan identitas, jalankan perintah berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan terkelola:
     - list-entities-for-policy
   - Untuk membuat daftar kebijakan terkelola yang terlampir pada identitas (pengguna, grup pengguna, atau peran):
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Untuk menyunting kebijakan terkelola pelanggan, jalankan perintah berikut:
   - create-policy-version
4. (Opsional) Untuk memvalidasi kebijakan terkelola pelanggan, jalankan perintah IAM Access Analyzer berikut:
   - validasi-kebijakan

Menyetel versi default kebijakan terkelola pelanggan (AWS CLI)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari AWS CLI.

Untuk mengatur versi default kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, jalankan perintah berikut:
   - daftar-kebijakan
2. Untuk mengatur versi default kebijakan terkelola pelanggan, jalankan perintah berikut:
   - set-default-policy-version

Menghapus versi kebijakan terkelola pelanggan ()AWS CLI

Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari AWS CLI.

Untuk menghapus versi kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, jalankan perintah berikut:
   - daftar-kebijakan
2. Untuk menghapus kebijakan terkelola pelanggan, jalankan perintah berikut:
   - delete-policy-version

Mengedit kebijakan sebaris ()AWS CLI

Anda dapat mengedit kebijakan inline dari AWS CLI.

Untuk mengedit kebijakan inline ()AWS CLI

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan sebaris yang terkait dengan identitas (pengguna, grup pengguna, atau peran):
     - list-user-policies
     - list-role-policies
     - list-group-policies
   - Untuk mengambil informasi rinci tentang kebijakan inline:

- get-user-policy
- get-role-policy
- get-group-policy

2. Untuk mengedit kebijakan inline, jalankan perintah berikut:

- put-user-policy
- put-role-policy
- put-group-policy

3. (Opsional) Untuk memvalidasi kebijakan inline, jalankan perintah IAM Access Analyzer berikut:

- validasi-kebijakan

Edit kebijakan IAM (AWS API)

Sebuahkebijakan adalah entitas yang, saat dilampirkan dengan sebuah identitas atau sumber daya, menjelaskan izinnya. Anda dapat menggunakan AWS API untuk mengedit kebijakan terkelola pelanggan dan kebijakan inline di IAM. AWS kebijakan terkelola tidak dapat diedit. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

Untuk informasi selengkapnya tentang struktur dan sintaks, lihat Kebijakan dan izin di AWS Identity and Access Management dan Referensi elemen kebijakan IAM JSON.

Prasyarat

Sebelum Anda mengubah izin untuk suatu kebijakan, Anda harus meninjau aktivitas tingkat layanannya yang terbaru. Ini penting karena Anda tidak ingin menghapus akses dari suatu prinsipal (orang atau aplikasi) yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Mengedit kebijakan terkelola pelanggan (AWS API)

Anda dapat mengedit kebijakan yang dikelola pelanggan menggunakan AWS API.

> **Note**
>
> Kebijakan terkelola dapat memiliki hingga lima versi. Apabila Anda perlu melakukan perubahan pada kebijakan terkelola pelanggan lebih dari lima versi, Anda harus terlebih dahulu menghapus satu atau beberapa versi yang ada.

Untuk mengedit kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh kembali informasi detail tentang kebijakan yang dikelola: GetPolicy
2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan terkelola:
     - ListEntitiesForPolicy
   - Untuk membuat daftar kebijakan terkelola yang terlampir pada identitas (pengguna, grup pengguna, atau peran):
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Untuk menyunting kebijakan terkelola pelanggan, hubungi layanan berikut:
   - CreatePolicyVersion
4. (Opsional) Untuk memvalidasi kebijakan terkelola pelanggan, hubungi operasi IAM Access Analyzer berikut:
   - ValidatePolicy

Menyetel versi default dari kebijakan terkelola pelanggan (AWS API)

Anda dapat menyetel versi default kebijakan terkelola pelanggan dari AWS API.

Untuk menyetel versi default kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:
   - ListPolicies
2. Untuk mengatur versi standar kebijakan terkelola pelanggan, hubungi layanan berikut:
   - SetDefaultPolicyVersion

Menghapus versi kebijakan terkelola pelanggan (AWS API)

Anda dapat menghapus versi kebijakan yang dikelola pelanggan dari AWS API.

Untuk menghapus versi kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk membuat daftar kebijakan terkelola, hubungi layanan berikut:
   - ListPolicies
2. Untuk menghapus kebijakan terkelola pelanggan, hubungi layanan berikut:
   - DeletePolicyVersion

Mengedit kebijakan sebaris (AWS API)

Anda dapat mengedit kebijakan inline dari AWS API.

Untuk mengedit kebijakan inline (AWS API)

1. (Opsional) Untuk melihat informasi tentang kebijakan sebaris, hubungi operasi berikut:
   - Untuk mencantumkan kebijakan sebaris yang terkait dengan identitas (pengguna, grup pengguna, atau peran):
     - ListUserPolicies
     - ListRolePolicies
     - ListGroupPolicies
   - Untuk mengambil informasi rinci tentang kebijakan inline:
     - GetUserPolicy
     - GetRolePolicy

- GetGroupPolicy

2. Untuk mengedit kebijakan inline, panggil operasi berikut:

   - PutUserPolicy
   - PutRolePolicy
   - PutGroupPolicy

3. (Opsional) Untuk memvalidasi kebijakan inline, jalankan operasi IAM Access Analyzer berikut:

   - ValidatePolicy

Hapus kebijakan IAM

Anda dapat menghapus kebijakan IAM menggunakan Konsol Manajemen AWS, AWS Command Line Interface (AWS CLI), atau AWS API.

> **Note**
>
> Penghapusan kebijakan IAM bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur dan sintaks kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management dan. Referensi elemen kebijakan IAM JSON

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Lihat informasi yang lebih lengkap di IAM dan AWS STS kuota.

Topik

- Hapus kebijakan IAM (konsol)
- Hapus kebijakan IAM ()AWS CLI
- Hapus kebijakan IAM (AWS API)

Hapus kebijakan IAM (konsol)

Anda dapat menggunakan Konsol Manajemen AWS untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline di IAM. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

> **Note**
>
> Penghapusan kebijakan IAM bersifat permanen. Setelah kebijakan dihapus, kebijakan tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur dan sintaks kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management dan. Referensi elemen kebijakan IAM JSON

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan IAM (konsol)

Anda mungkin perlu menghapus kebijakan terkelola pelanggan ketika sudah usang atau tidak lagi selaras dengan persyaratan keamanan organisasi dan kebutuhan kontrol akses. Dengan menghapus kebijakan yang tidak perlu, Anda mengurangi potensi risiko keamanan yang terkait dengan kebijakan yang sudah ketinggalan zaman atau tidak terpakai. Anda dapat menghapus kebijakan yang dikelola pelanggan untuk menghapusnya dari kebijakan Anda Akun AWS. Anda tidak dapat menghapus kebijakan AWS terkelola.

Console

Untuk menghapus kebijakan terkelola pelanggan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.

3. Pilih tombol radio di sebelah kebijakan yang dikelola pelanggan untuk dihapus. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Tindakan, lalu pilih Hapus.
5. Ikuti petunjuk untuk mengonfirmasi bahwa Anda ingin menghapus kebijakan, lalu pilih Hapus.

Menghapus kebijakan sebaris (konsol)

Anda mungkin perlu menghapus kebijakan sebaris ketika izin khusus yang diberikannya tidak lagi diperlukan untuk pengguna, grup, atau peran IAM yang dilampirkan langsung. Menghapus kebijakan inline yang tidak perlu membantu mengurangi risiko akses yang tidak diinginkan, terutama karena kebijakan inline tidak dapat digunakan kembali atau dibagikan di beberapa identitas seperti kebijakan terkelola. Anda dapat menghapus kebijakan inline untuk menghapusnya dari kebijakan Anda Akun AWS. Anda tidak dapat menghapus kebijakan terkelola AWS .

Console

Untuk menghapus kebijakan inline untuk pengguna, grup, atau peran IAM

1. Di panel navigasi, pilih User groups (Grup pengguna), Users (Pengguna), atau Roles (Peran).
2. Pilih nama grup pengguna, pengguna, atau peran dengan kebijakan yang ingin Anda hapus. Kemudian pilih tab Perizinan.
3. Pilih kotak centang di samping kebijakan yang akan dihapus dan pilih Hapus. Kemudian, dalam dialog konfirmasi, konfirmasikan penghapusan dan penghapusan kebijakan.
   - Untuk menghapus kebijakan sebaris di Pengguna atau Peran, pilih Hapus untuk mengonfirmasi penghapusan.
   - Jika Anda menghapus kebijakan inline tunggal di User groups (Pengguna grup), ketik nama kebijakan dan pilih Delete (Hapus). Jika Anda menghapus beberapa kebijakan inline diPengguna grup, ketik jumlah kebijakan yang Anda hapus diikuti oleh **`inline policies`**, lalu pilih Delete (Hapus). Sebagai contoh, jika Anda menghapus tiga kebijakan inline, ketik **`3 inline policies`**.

Hapus kebijakan IAM ()AWS CLI

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline di IAM. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

> **Note**
>
> Penghapusan kebijakan IAM bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur dan sintaks kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management dan. Referensi elemen kebijakan IAM JSON

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan terkelola pelanggan ()AWS CLI

Anda dapat menghapus kebijakan terkelola pelanggan dari AWS Command Line Interface.

Untuk menghapus kebijakan terkelola pelanggan (AWS CLI)

1. (Opsional) Untuk melihat informasi tentang suatu kebijakan, jalankan perintah berikut:
   - Untuk mencantumkan kebijakan terkelola: daftar-kebijakan
   - Untuk memperoleh informasi terperinci tentang kebijakan terkelola: dapatkan-kebijakan
2. (Opsional) Untuk mengetahui tentang hubungan antara kebijakan dan identitas, jalankan perintah berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) tempat kebijakan terkelola dilampirkan, jalankan perintah berikut:
     - list-entities-for-policy
   - Untuk mencantumkan kebijakan terkelola yang terlampir pada sebuah identitas (pengguna, grup pengguna, atau peran), jalankan salah satu perintah berikut:
     - list-attached-user-policies
     - list-attached-group-policies

- list-attached-role-policies

3. Untuk menghapus kebijakan terkelola pelanggan, jalankan perintah berikut:

   - menghapus-kebijakan

Menghapus kebijakan inline ()AWS CLI

Anda dapat menghapus kebijakan inline dari. AWS CLI

Untuk menghapus kebijakan inline (AWS CLI)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies

2. (Opsional) Untuk mengambil dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), gunakan salah satu perintah berikut:

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. Untuk menghapus kebijakan inline dari identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), gunakan salah satu perintah berikut:

   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Hapus kebijakan IAM (AWS API)

Anda dapat menggunakan AWS API untuk menghapus kebijakan terkelola pelanggan dan kebijakan inline di IAM. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

> **Note**
>
> Penghapusan kebijakan IAM bersifat permanen. Setelah kebijakan dihapus, kebijakan tersebut tidak dapat dipulihkan.

Untuk informasi selengkapnya tentang struktur dan sintaks kebijakan IAM, lihat Kebijakan dan izin di AWS Identity and Access Management dan. Referensi elemen kebijakan IAM JSON

Untuk informasi lebih lanjut tentang perbedaan antara kebijakan terkelola dan kebijakan inline, lihat Kebijakan terkelola dan kebijakan inline.

Prasyarat

Sebelum menghapus suatu kebijakan, Anda harus meninjau aktivitas tingkat-layanan terakhirnya. Ini penting karena Anda tidak ingin menghapus akses dari (orang atau aplikasi) utama yang menggunakannya. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Menghapus kebijakan terkelola pelanggan (AWS API)

Anda dapat menghapus kebijakan terkelola pelanggan menggunakan AWS API.

Untuk menghapus kebijakan terkelola pelanggan (AWS API)

1. (Opsional) Untuk melihat informasi mengenai kebijakan, hubungi layanan berikut:
   - Untuk mencantumkan kebijakan terkelola: ListPolicies
   - Untuk memperoleh kembali informasi detail tentang kebijakan yang dikelola: GetPolicy
2. (Opsional) Untuk mengetahui perihal hubungan antara kebijakan dan identitas, hubungi layanan berikut:
   - Untuk mencantumkan identitas (pengguna IAM, grup IAM, dan peran IAM) yang dilampirkan kebijakan terkelola, panggil operasi berikut:
     - ListEntitiesForPolicy
   - Untuk mencantumkan kebijakan terkelola pada suatu identitas (pengguna, grup pengguna, atau peran), panggil salah satu operasi berikut:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies

- ListAttachedRolePolicies

3. Untuk menghapus kebijakan terkelola pelanggan, hubungi layanan berikut:

   - DeletePolicy

Menghapus kebijakan inline (API)AWS

Anda dapat menghapus kebijakan inline menggunakan AWS API.

Untuk menghapus kebijakan inline (AWS API)

1. (Opsional) Untuk mendaftar seluruh kebijakan inline yang dilampirkan pada identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (Opsional) Untuk memperoleh dokumen kebijakan inline yang disematkan di identitas (pengguna, grup pengguna, atau peran), hubungi salah satu layanan berikut:

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

3. Untuk menghapus kebijakan inline dari suatu identitas (pengguna, grup pengguna, atau peran yang bukan peran terkait layanan), hubungi salah satu layanan berikut:

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses

Sebagai administrator, Anda dapat memberikan izin ke sumber daya IAM (peran, pengguna, grup pengguna, atau kebijakan) di luar yang mereka butuhkan. IAM menyediakan informasi yang terakhir diakses untuk membantu Anda mengidentifikasi izin yang tidak digunakan sehingga Anda dapat menghapusnya. Anda dapat menggunakan informasi yang diakses terakhir untuk menyempurnakan

kebijakan Anda dan mengizinkan akses hanya ke layanan dan tindakan yang digunakan identitas dan kebijakan IAM Anda. Ini membantu Anda untuk lebih menganut praktik terbaik dengan privilese paling sedikit. Anda dapat melihat informasi yang terakhir diakses untuk identitas atau kebijakan yang ada di IAM atau. AWS Organizations

Anda dapat terus memantau informasi yang diakses terakhir dengan penganalisis akses yang tidak digunakan. Untuk informasi selengkapnya, lihat Temuan untuk akses eksternal dan tidak terpakai.

Topik

- Jenis informasi yang terakhir diakses untuk IAM
- Informasi terakhir yang diakses untuk AWS Organizations
- Hal yang perlu diketahui tentang informasi yang terakhir diakses
- Izin diperlukan
- Memecahkan masalah aktivitas untuk IAM dan entitas AWS Organizations
- Tempat AWS melacak informasi terakhir yang diakses
- Lihat informasi yang terakhir diakses untuk IAM
- Lihat informasi terakhir yang diakses untuk AWS Organizations
- Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses
- Tindakan IAM terakhir mengakses layanan dan tindakan informasi

Jenis informasi yang terakhir diakses untuk IAM

Anda dapat melihat dua jenis informasi terakhir yang diakses untuk identitas IAM: informasi AWS layanan yang diizinkan dan informasi tindakan yang diizinkan. Informasi tersebut mencakup tanggal dan waktu ketika upaya untuk mengakses AWS API dilakukan. Untuk tindakan, informasi yang terakhir diakses melaporkan tindakan manajemen layanan. Tindakan manajemen meliputi pembuatan, penghapusan, dan tindakan modifikasi. Untuk mempelajari lebih lanjut tentang cara melihat informasi yang terakhir diakses untuk IAM, lihat Lihat informasi yang terakhir diakses untuk IAM.

Misalnya skenario untuk menggunakan informasi yang diakses terakhir untuk membuat keputusan tentang izin yang Anda berikan ke identitas IAM Anda, lihat. Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses

Untuk mempelajari lebih lanjut tentang bagaimana tersedianya informasi untuk tindakan manajemen, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.

Informasi terakhir yang diakses untuk AWS Organizations

Jika Anda masuk menggunakan kredensi akun manajemen, Anda dapat melihat informasi layanan yang terakhir diakses untuk AWS Organizations entitas atau kebijakan di organisasi Anda. AWS Organizations entitas termasuk akar organisasi, unit organisasi (OUs), atau akun. Informasi terakhir yang diakses untuk AWS Organizations mencakup informasi tentang layanan yang diizinkan oleh kebijakan kontrol layanan (SCP). Informasi menunjukkan prinsip mana (pengguna root, pengguna IAM, atau peran) dalam organisasi atau akun yang terakhir mencoba mengakses layanan dan kapan. Untuk mempelajari lebih lanjut tentang laporan dan cara melihat informasi yang terakhir diakses AWS Organizations, lihatLihat informasi terakhir yang diakses untuk AWS Organizations.

Misalnya skenario untuk menggunakan informasi yang diakses terakhir untuk membuat keputusan tentang izin yang Anda berikan kepada AWS Organizations entitas Anda, lihatContoh alur perencanaan untuk menggunakan informasi yang terakhir diakses.

Hal yang perlu diketahui tentang informasi yang terakhir diakses

Sebelum Anda menggunakan informasi yang terakhir diakses dari laporan untuk mengubah izin identitas atau AWS Organizations entitas IAM, tinjau detail berikut tentang informasi tersebut.

- Periode pelacakan - Aktivitas terbaru muncul di konsol IAM dalam waktu empat jam. Periode pelacakan untuk informasi layanan setidaknya 400 hari tergantung pada kapan layanan mulai melacak informasi tindakan. Periode pelacakan untuk informasi tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk tindakan Amazon EC2, IAM, dan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan lainnya dimulai pada 23 Mei 2023. Untuk daftar layanan yang informasi terakhir diakses tindakan yang tersedia, lihatTindakan IAM terakhir mengakses layanan dan tindakan informasi. Untuk informasi selengkapnya tentang tindakan Wilayah mana yang terakhir diakses informasi tersedia, lihatTempat AWS melacak informasi terakhir yang diakses.
- Upaya dilaporkan — Data layanan yang terakhir diakses mencakup semua upaya untuk mengakses AWS API, bukan hanya upaya yang berhasil. Ini termasuk semua upaya yang dilakukan menggunakan Konsol Manajemen AWS, AWS API melalui salah satu SDKs, atau salah satu alat baris perintah. Entri tak terduga dalam layanan data terakhir yang diakses tidak berarti akun Anda telah disusupi, karena permintaan tersebut mungkin ditolak. Lihat CloudTrail log Anda sebagai sumber otoritatif untuk informasi tentang semua panggilan API dan apakah mereka berhasil atau ditolak aksesnya.
- PassRole— `iam:PassRole` Tindakan tidak dilacak dan tidak termasuk dalam tindakan IAM informasi terakhir diakses.

- Action last access information — Action last access information tersedia untuk tindakan manajemen layanan yang diakses oleh identitas IAM. Lihat daftar layanan dan tindakan mereka untuk tindakan yang terakhir diakses melaporkan informasi.

  > **Note**
  >
  > Informasi tindakan yang terakhir diakses tidak tersedia untuk peristiwa pesawat data apa pun.

- Peristiwa manajemen - IAM menyediakan informasi tindakan untuk peristiwa manajemen layanan yang dicatat oleh CloudTrail. Terkadang, peristiwa manajemen CloudTrail juga disebut operasi kontrol pesawat atau peristiwa kontrol pesawat. Acara manajemen memberikan visibilitas ke dalam operasi administratif yang dilakukan pada sumber daya di Anda Akun AWS. Untuk mempelajari selengkapnya tentang peristiwa manajemen di CloudTrail, lihat peristiwa pengelolaan log di Panduan AWS CloudTrail Pengguna.
- Laporkan pemilik – Hanya penanggung jawab yang membuat laporan yang dapat melihat detail laporan. Ini berarti bahwa ketika Anda melihat informasi di Konsol Manajemen AWS, Anda mungkin harus menunggu untuk menghasilkan dan memuat. Jika Anda menggunakan AWS API AWS CLI atau untuk mendapatkan detail laporan, kredensional Anda harus sesuai dengan kredensi prinsipal yang menghasilkan laporan. Jika Anda menggunakan kredensi sementara untuk peran atau prinsipal pengguna AWS STS gabungan, Anda harus membuat dan mengambil laporan selama sesi yang sama. Untuk informasi lebih lanjut tentang penanggung jawab sesi peran yang diasumsikan, lihat AWS Elemen kebijakan JSON: Principal.
- Sumber daya IAM — Informasi terakhir yang diakses untuk IAM mencakup sumber daya IAM (peran, pengguna, grup IAM, dan kebijakan) di akun Anda. Informasi yang terakhir diakses untuk AWS Organizations menyertakan prinsipal (pengguna IAM, peran IAM, atau Pengguna root akun AWS) dalam entitas yang ditentukan. AWS Organizations Informasi terakhir yang diakses tidak termasuk upaya yang tidak diautentikasi.
- Jenis kebijakan IAM — Informasi terakhir yang diakses untuk IAM mencakup layanan yang diizinkan oleh kebijakan identitas IAM. Ini merupakan kebijakan yang melekat pada suatu peran atau terlampir pada pengguna secara langsung atau melalui grup. Akses yang diperbolehkan oleh jenis kebijakan lain tidak termasuk dalam laporan Anda. Jenis kebijakan yang dikecualikan mencakup kebijakan berbasis sumber daya, daftar kontrol akses, batas izin IAM AWS Organizations SCPs, dan kebijakan sesi. Izin yang diberikan oleh peran yang terkait dengan layanan ditentukan oleh layanan yang terkait dan tidak dapat dimodifikasi di IAM. Untuk mempelajari lebih lanjut tentang peran yang berkaitan dengan layanan, lihat Buat peran tertaut

layanan Untuk mempelajari cara jenis kebijakan yang berbeda dievaluasi untuk memungkinkan atau menolak akses, lihat Logika evaluasi kebijakan.

- AWS Organizations Jenis kebijakan — Informasi untuk hanya AWS Organizations mencakup layanan yang diizinkan oleh kebijakan kontrol layanan warisan AWS Organizations entitas (SCPs). SCPs adalah kebijakan yang dilampirkan ke root, OU, atau akun. Akses yang diperbolehkan oleh jenis kebijakan lain tidak termasuk dalam laporan Anda. Jenis kebijakan yang dikecualikan meliputi kebijakan berbasis identitas, kebijakan berbasis sumber daya, access control list, batasan izin IAM, dan kebijakan sesi. Untuk mempelajari cara berbagai jenis kebijakan dievaluasi untuk memungkinkan atau menolak akses, lihat Logika evaluasi kebijakan.
- Menentukan ID kebijakan — Saat Anda menggunakan AWS CLI atau AWS API untuk membuat laporan untuk informasi yang terakhir diakses AWS Organizations, Anda dapat menentukan ID kebijakan secara opsional. Laporan yang dihasilkan mencakup informasi untuk layanan yang diperbolehkan hanya oleh kebijakan tersebut. Informasi tersebut mencakup aktivitas akun terbaru di AWS Organizations entitas tertentu atau anak entitas. Untuk informasi selengkapnya, lihat aws iam generate-organizations-access-report atau GenerateOrganizationsAccessReport.
- AWS Organizations akun manajemen — Anda harus masuk ke akun manajemen organisasi Anda untuk melihat informasi layanan yang terakhir diakses. Anda dapat memilih untuk melihat informasi untuk akun manajemen menggunakan konsol IAM, API AWS CLI, atau AWS API. Laporan yang dihasilkan mencantumkan semua AWS layanan, karena akun manajemen tidak dibatasi oleh SCPs. Jika Anda menentukan ID kebijakan di CLI atau API, kebijakan tersebut akan diabaikan. Untuk setiap layanan, laporan mencakup informasi hanya untuk akun manajemen. Namun, laporan untuk AWS Organizations entitas lain tidak mengembalikan informasi untuk aktivitas di akun manajemen.
- AWS Organizations pengaturan — Administrator harus mengaktifkan SCPs di root organisasi Anda sebelum Anda dapat menghasilkan data untuk AWS Organizations.

Izin diperlukan

Untuk melihat informasi yang terakhir diakses di Konsol Manajemen AWS, Anda harus memiliki kebijakan yang memberikan izin yang diperlukan.

Izin untuk informasi IAM

Untuk menggunakan konsol IAM guna melihat informasi terakhir yang diakses pengguna, peran, atau kebijakan IAM, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- `iam:GenerateServiceLastAccessedDetails`

- `iam:Get*`
- `iam:List*`

Izin ini memungkinkan pengguna melihat hal berikut:

- Pengguna, kelompok, atau peran yang dilampirkan pada kebijakan terkelola
- Layanan yang dapat diakses pengguna atau peran
- Terakhir kali mereka mengakses layanan
- Terakhir kali mereka mencoba menggunakan tindakan Amazon EC2, IAM, Lambda, atau Amazon S3 tertentu

Untuk menggunakan AWS API AWS CLI atau untuk melihat informasi terakhir yang diakses untuk IAM, Anda harus memiliki izin yang cocok dengan operasi yang ingin Anda gunakan:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan melihat informasi terakhir yang diakses IAM. Selain itu, ini memungkinkan akses baca-saja ke seluruh IAM. Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

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

```
}
```

Izin untuk informasi AWS Organizations

Untuk menggunakan konsol IAM untuk melihat laporan untuk entitas root, OU, atau akun di AWS Organizations, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Untuk menggunakan AWS CLI atau AWS API untuk melihat layanan informasi yang terakhir diakses AWS Organizations, Anda harus memiliki kebijakan yang mencakup tindakan berikut:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan layanan melihat informasi yang terakhir diakses. AWS Organizations Selain itu, ini

memungkinkan akses hanya-baca ke semua. AWS Organizations Kebijakan ini menetapkan izin untuk akses terprogram dan konsol.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

Anda juga dapat menggunakan kunci OrganizationsPolicyId kondisi iam: untuk memungkinkan pembuatan laporan hanya untuk AWS Organizations kebijakan tertentu. Untuk contoh kebijakan, lihat IAM: Lihat informasi layanan yang terakhir diakses untuk kebijakan AWS Organizations.

Memecahkan masalah aktivitas untuk IAM dan entitas AWS Organizations

Dalam beberapa kasus, tabel informasi Konsol Manajemen AWS terakhir Anda yang diakses mungkin kosong. Atau mungkin permintaan Anda AWS CLI atau AWS API mengembalikan kumpulan informasi kosong atau bidang nol. Dalam kejadian ini, tinjau masalah berikut:

- Untuk informasi tindakan yang terakhir kali diakses, tindakan yang Anda harapkan untuk dilihat mungkin tidak akan dikembalikan dalam daftar. Hal ini dapat terjadi baik karena identitas IAM tidak memiliki izin untuk tindakan, atau AWS belum melacak tindakan untuk informasi yang terakhir diakses.
- Untuk pengguna IAM, pastikan bahwa pengguna memiliki sekurangnya satu kebijakan selaras atau terkelola terlampir, baik langsung atau melalui keanggotaan grup.
- Untuk grup IAM, verifikasi bahwa grup tersebut memiliki sekurangnya satu kebijakan selaras atau terkelola terlampir.

- Untuk grup IAM, laporan hanya mengembalikan informasi layanan yang terakhir diakses bagi anggota yang menggunakan kebijakan grup tersebut untuk mengakses layanan. Untuk mempelajari apakah anggota menggunakan kebijakan lain, tinjau informasi yang terakhir diakses untuk pengguna tersebut.
- Untuk peran IAM, verifikasi bahwa grup tersebut memiliki sekurangnya satu kebijakan selaras atau terkelola terlampir.
- Untuk entitas IAM (pengguna atau peran), tinjau jenis kebijakan lain yang mungkin memengaruhi izin entitas tersebut. Ini termasuk kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas izin IAM, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Evaluasi kebijakan untuk permintaan dalam satu akun.
- Untuk kebijakan IAM, pastikan kebijakan pengelolaan yang ditentukan dilampirkan pada setidaknya suatu pengguna, grup dengan anggota, atau peran.
- Untuk AWS Organizations entitas (root, OU, atau akun), pastikan Anda ditandatangani menggunakan kredensi akun AWS Organizations manajemen.
- Verifikasi bahwa SCPs aktif di root organisasi Anda.
- Informasi tindakan yang terakhir diakses hanya tersedia untuk tindakan yang tercantum dalamTindakan IAM terakhir mengakses layanan dan tindakan informasi.

Saat Anda membuat perubahan, tunggu setidaknya empat jam agar aktivitas muncul di laporan konsol IAM Anda. Jika Anda menggunakan AWS API AWS CLI atau, Anda harus membuat laporan baru untuk melihat informasi yang diperbarui.

Tempat AWS melacak informasi terakhir yang diakses

AWS mengumpulkan informasi yang terakhir diakses untuk AWS Wilayah standar. Saat AWS menambahkan Wilayah tambahan, Wilayah tersebut ditambahkan ke tabel berikut, termasuk tanggal yang AWS mulai melacak informasi di setiap Wilayah.

- Informasi layanan — Periode pelacakan untuk layanan setidaknya 400 hari, atau kurang jika Wilayah Anda mulai melacak fitur ini dalam 400 hari terakhir.
- Informasi tindakan – Periode pelacakan untuk tindakan manajemen Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk tindakan manajemen Amazon EC2, IAM, dan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk tindakan manajemen semua layanan lainnya dimulai pada 23 Mei 2023. Jika tanggal pelacakan suatu Wilayah lebih lambat dari 23 Mei 2023, maka tindakan yang terakhir diakses informasi dari Wilayah tersebut akan dimulai di kemudian hari.

Nama wilayah	Region	Tanggal mulai pelacakan
AS Timur (Ohio)	as-timur-2	27 Oktober 2017
AS Timur (Virginia Utara)	as-timur-1	1 Oktober 2015
AS Barat (California Utara)	as-barat-1	1 Oktober 2015
AS Barat (Oregon)	as-barat-2	1 Oktober 2015
Africa (Cape Town)	af-selatan-1	22 April 2020
Asia Pasifik (Hong Kong)	ap-timur-1	24 April 2019
Asia Pasifik (Hyderabad)	ap-south-2	22 November 2022
Asia Pasifik (Jakarta)	ap-southeast-3	13 Desember 2021
Asia Pacific (Melbourne)	ap-southeast-4	23 Januari 2023
Asia Pasifik (Mumbai)	ap-selatan-1	27 Juni 2016
Asia Pacific (Osaka)	ap-northeast-3	Februari 11, 2018
Asia Pasifik (Seoul)	ap-timur laut-2	6 Januari 2016
Asia Pasifik (Singapura)	ap-tenggara-1	1 Oktober 2015
Asia Pasifik (Sydney)	ap-tenggara-2	1 Oktober 2015
Asia Pasifik (Tokyo)	ap-timur laut-1	1 Oktober 2015
Kanada (Pusat)	ca-sentral-1	28 Oktober 2017
Eropa (Frankfurt)	eu-sentral-1	1 Oktober 2015
Eropa (Irlandia)	eu-barat-1	1 Oktober 2015
Eropa (London)	eu-barat-2	28 Oktober 2017
Eropa (Milan)	eu-selatan-1	28 April 2020

Nama wilayah	Region	Tanggal mulai pelacakan
Eropa (Paris)	eu-barat-3	18 Desember 2017
Eropa (Spanyol)	eu-south-2	15 November 2022
Eropa (Stockholm)	eu-utara-1	12 Desember 2018
Europe (Zurich)	eu-central-2	8 November 2022
Israel (Tel Aviv)	il-central-1	1 Agustus 2023
Timur Tengah (Bahrain)	me-selatan-1	29 Juli 2019
Timur Tengah (UAE)	me-central-1	30 Agustus 2022
Amerika Selatan (Sao Paulo)	sa-timur-1	11 Desember 2015
AWS GovCloud (AS-Timur)	us-gov-east-1	Juli 1, 2023
AWS GovCloud (AS-Barat)	us-gov-west-1	Juli 1, 2023

Jika region tidak tercantum dalam tabel sebelumnya, maka region tersebut belum memberikan informasi yang terakhir diakses.

AWS Wilayah adalah kumpulan AWS sumber daya di wilayah geografis. Region-region dikelompokkan menjadi beberapa bagian. Region standar adalah region yang termasuk ke bagian `aws`. Untuk informasi selengkapnya tentang partisi yang berbeda, lihat Format Nama Sumber Daya Amazon (ARNs) di. Referensi Umum AWS Untuk informasi selengkapnya tentang Wilayah, lihat Tentang AWS Wilayah juga di Referensi Umum AWS.

Lihat informasi yang terakhir diakses untuk IAM

Anda dapat melihat informasi terakhir yang diakses untuk IAM menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API. Lihat daftar layanan dan tindakan mereka yang informasi terakhir diakses ditampilkan. Untuk informasi selengkapnya perihal melihat informasi yang terakhir diakses, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Anda dapat melihat informasi untuk jenis sumber daya berikut di IAM. Dalam setiap kejadian, informasi tersebut mencakup layanan yang diizinkan untuk periode pelaporan tertentu:

- Pengguna — Lihat terakhir kali pengguna mencoba mengakses setiap layanan yang diizinkan.
- Grup pengguna — Melihat informasi tentang terakhir kali anggota grup pengguna mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total anggota yang mencoba mengaksesnya.
- Peran – Lihat terakhir kali seseorang menggunakan peran tersebut dalam upaya mengakses setiap layanan yang diizinkan.
- Kebijakan – Melihat informasi terakhir kali pengguna atau peran mencoba mengakses setiap layanan yang diizinkan. Laporan ini juga mencakup jumlah total entitas yang mencoba mengaksesnya.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk sumber daya di IAM, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk informasi Anda. Untuk lebih detailnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Melihat informasi untuk IAM (konsol)

Anda dapat melihat informasi terakhir yang diakses untuk IAM pada tab Terakhir Diakses di konsol IAM.

Melihat informasi untuk IAM (konsol)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Grup pengguna, Pengguna, Peran, atau Kebijakan.
3. Pilih nama pengguna, grup pengguna, peran, atau kebijakan apa pun untuk membuka halaman Ringkasannya dan pilih tab Terakhir Diakses. Lihat informasi berikut, berdasarkan sumber daya yang Anda pilih:

   - Grup pengguna - Lihat daftar layanan yang dapat diakses oleh anggota grup pengguna. Anda juga dapat melihat kapan anggota terakhir mengakses layanan, kebijakan grup pengguna apa yang mereka gunakan, dan anggota grup pengguna mana yang membuat permintaan. Pilih nama kebijakan untuk dipelajari baik merupakan kebijakan terkelola atau kebijakan grup

pengguna inline. Pilih nama anggota grup pengguna untuk melihat semua anggota grup pengguna dan kapan mereka terakhir kali mengakses layanan.

- Pengguna – Lihat daftar layanan yang dapat diakses pengguna. Anda juga dapat melihat kapan mereka terakhir mengakses layanan, dan kebijakan apa yang saat ini terkait dengan pengguna. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola, kebijakan pengguna inline, atau kebijakan inline untuk grup pengguna.
- Peran – Lihat daftar layanan yang dapat diakses oleh peran, kapan peran terakhir mengakses layanan, dan kebijakan apa yang digunakan. Pilih nama kebijakan untuk dipelajari baik ia merupakan kebijakan terkelola atau kebijakan peran selaras.
- Kebijakan – Lihat daftar layanan dengan tindakan yang diperbolehkan dalam kebijakan. Anda juga dapat melihat kapan kebijakan terakhir digunakan untuk mengakses layanan, dan entitas (pengguna atau peran) mana yang menggunakan kebijakan tersebut. Tanggal Terakhir diakses juga mencakup kapan akses diberikan ke kebijakan ini melalui kebijakan lain. Pilih nama entitas untuk dipelajari yang sudah terlampirkan dengan kebijakan ini dan kapan terakhir kali mereka mengakses layanan.

4. Di kolom Layanan tabel, pilih nama salah satu layanan yang menyertakan informasi tindakan yang terakhir diakses untuk melihat daftar tindakan manajemen yang telah dicoba diakses oleh entitas IAM. Anda dapat melihat Wilayah AWS dan stempel waktu yang menunjukkan kapan seseorang terakhir mencoba melakukan tindakan.
5. Kolom Terakhir diakses ditampilkan untuk layanan dan tindakan manajemen layanan yang menyertakan informasi tindakan yang terakhir diakses. Tinjau hasil-hasil berikut yang kemungkinan muncul dalam kolom ini. Hasil ini bervariasi tergantung pada apakah layanan atau tindakan diizinkan, diakses, dan apakah itu dilacak oleh AWS untuk informasi yang terakhir diakses.

   <jumlah> hari yang lalu

   Jumlah hari sejak layanan atau tindakan digunakan dalam periode pelacakan. Periode pelacakan layanan adalah selama 400 hari terakhir. Periode pelacakan untuk tindakan Amazon S3 dimulai pada 12 April 2020. Periode pelacakan untuk tindakan Amazon EC2, IAM, dan Lambda dimulai pada 7 April 2021. Periode pelacakan untuk semua layanan lainnya dimulai pada 23 Mei 2023. Untuk mempelajari selengkapnya tentang melacak tanggal mulai untuk masing-masing Wilayah AWS, lihatTempat AWS melacak informasi terakhir yang diakses.

Tidak diakses dalam periode pelacakan

Layanan atau tindakan yang dilacak belum digunakan oleh entitas dalam periode pelacakan.

Anda dapat memiliki izin untuk tindakan yang tidak muncul dalam daftar. Ini dapat terjadi jika informasi pelacakan untuk tindakan saat ini tidak disertakan oleh AWS. Anda tidak diperkenankan mengambil keputusan izin hanya berdasarkan ketiadaan informasi pelacakan. Alih-alih, kami menyarankan agar Anda menggunakan informasi ini untuk menginformasikan dan mendukung strategi Anda secara keseluruhan untuk memberikan privilese paling sedikit. Periksa kebijakan Anda untuk mengonfirmasi bahwa tingkat akses sesuai.

Melihat informasi untuk IAM ()AWS CLI

Anda dapat menggunakan AWS CLI untuk mengambil informasi tentang terakhir kali sumber daya IAM digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, Amazon EC2, IAM, dan Lambda. Sumber daya IAM dapat berupa pengguna, grup pengguna, peran, atau kebijakan.

Untuk melihat informasi untuk IAM ()AWS CLI

1. Buat laporan. Permintaan harus menyertakan ARN sumber daya IAM (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat detail akses baik untuk layanan maupun layanan beserta tindakan. Permintaan tersebut menghasilkan `job-id` yang kemudian dapat Anda gunakan di operasi `get-service-last-accessed-details` dan `get-service-last-accessed-details-with-entities` untuk memonitor `job-status` hingga pekerjaan selesai.

   - aws iam generate-service-last-accessed -detail
2. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

   - aws iam get-service-last-accessed -detail

   Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `generate-service-last-accessed-details` operasi:

- Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir pengguna dan ARN pengguna.
- Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ia juga menyatakan ARN pengguna tersebut dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntities untuk memperoleh daftar semua anggota.
- Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir peran dan ARN peran.
- Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna atau peran) untuk mengakses layanan menggunakan kebijakan. Ia juga menyatakan ARN dari entitas tersebut dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini menghasilkan daftar entitas setiap ARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir kali mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - aws iam get-service-last-accessed - details-with-entities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas izin IAM, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Evaluasi kebijakan untuk permintaan dalam satu akun.

   - aws iam list-policies-granting-service -akses

Melihat informasi untuk IAM (AWS API)

Anda dapat menggunakan AWS API untuk mengambil informasi tentang terakhir kali sumber daya IAM digunakan untuk mencoba mengakses AWS layanan dan tindakan Amazon S3, Amazon EC2, IAM, dan Lambda. Sumber daya IAM dapat berupa pengguna, grup pengguna, peran, atau kebijakan. Anda dapat menentukan tingkat perincian yang ingin Anda buat dalam laporan untuk melihat baik layanan maupun layanan beserta tindakan.

Untuk melihat informasi untuk IAM (AWS API)

1. Buat laporan. Permintaan harus menyertakan ARN sumber daya IAM (pengguna, grup pengguna, peran, atau kebijakan) yang Anda inginkan laporannya. Ia menghasilkan `JobId` yang kemudian dapat Anda gunakan di operasi `GetServiceLastAccessedDetails` dan `GetServiceLastAccessedDetailsWithEntities` untuk memonitor `JobStatus` hingga pekerjaan selesai.

   - GenerateServiceLastAccessedDetails
2. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetServiceLastAccessedDetails

   Operasi ini menghasilkan informasi berikut, berdasarkan jenis sumber daya dan tingkat IT yang Anda minta di `GenerateServiceLastAccessedDetails` operasi:

   - Pengguna – Menghasilkan daftar layanan yang dapat diakses oleh pengguna tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir pengguna dan ARN pengguna.
   - Grup pengguna — Mengembalikan daftar layanan yang dapat diakses oleh anggota grup pengguna tertentu menggunakan kebijakan yang dilampirkan ke grup pengguna. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir yang dilakukan oleh anggota grup pengguna mana pun. Ia juga menyatakan ARN pengguna tersebut dan jumlah total anggota grup pengguna yang telah mencoba mengakses layanan. Gunakan GetServiceLastAccessedDetailsWithEntities untuk memperoleh daftar semua anggota.
   - Peran – Menghasilkan daftar layanan yang dapat diakses oleh peran tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu upaya terakhir peran dan ARN peran.
   - Kebijakan – Menghasilkan daftar layanan yang dapat diakses oleh kebijakan tertentu. Untuk setiap layanan, operasi menyatakan tanggal dan waktu percobaan terakhir entitas (pengguna

atau peran) untuk mengakses layanan menggunakan kebijakan. Ia juga menyatakan ARN dari entitas tersebut dan jumlah total entitas yang mencoba mengakses.

3. Pelajari lebih lanjut tentang entitas yang menggunakan izin grup pengguna atau kebijakan dalam upaya mengakses layanan tertentu. Operasi ini menghasilkan daftar entitas setiap ARN, ID, nama, jalur, jenis (pengguna atau peran), dan kapan mereka terakhir kali mencoba mengakses layanan. Anda juga dapat menggunakan operasi ini untuk pengguna dan peran, tetapi ia hanya menyatakan informasi tentang entitas tersebut.

   - GetServiceLastAccessedDetailsWithEntities

4. Pelajari lebih lanjut tentang kebijakan berbasis identitas yang digunakan oleh suatu identitas (pengguna, grup pengguna, atau peran) dalam upaya mengakses layanan tertentu. Saat Anda menetapkan identitas dan layanan, operasi ini menghasilkan daftar kebijakan izin yang dapat digunakan oleh identitas tersebut untuk mengakses layanan tertentu. Operasi ini memberikan status kebijakan terkini dan tidak bergantung pada laporan yang dihasilkan. Ini juga tidak menampilkan jenis kebijakan lain, seperti kebijakan berbasis sumber daya, daftar kontrol akses, kebijakan, batas izin IAM, atau AWS Organizations kebijakan sesi. Untuk informasi selengkapnya, lihat Jenis kebijakan atau Evaluasi kebijakan untuk permintaan dalam satu akun.

   - ListPoliciesGrantingServiceAccess

Lihat informasi terakhir yang diakses untuk AWS Organizations

Anda dapat melihat informasi layanan yang terakhir diakses untuk AWS Organizations menggunakan konsol IAM, AWS CLI, atau AWS API. Untuk informasi penting tentang data, izin yan diperlukan, pemecahan masalah, dan region yang didukung lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Saat masuk ke konsol IAM menggunakan kredentif akun AWS Organizations manajemen, Anda dapat melihat informasi untuk entitas apa pun di organisasi Anda. AWS Organizations entitas termasuk root organisasi, unit organisasi (OU), dan akun. Anda juga dapat menggunakan konsol IAM untuk melihat informasi kebijakan kontrol layanan (SCP) dalam organisasi Anda. IAM menunjukkan daftar layanan yang diizinkan oleh SCP apa pun yang berlaku untuk entitas. Untuk setiap layanan, Anda dapat melihat informasi aktivitas akun terbaru untuk AWS Organizations entitas yang dipilih atau anak entitas.

Saat Anda menggunakan AWS API AWS CLI or dengan kredentif akun manajemen, Anda dapat membuat laporan untuk entitas atau kebijakan apa pun di organisasi Anda. Laporan terprogram untuk suatu entitas mencakup daftar layanan yang diizinkan oleh SCP yang berlaku bagi entitas tersebut.

Untuk setiap layanannya, laporan ini mencakup aktivitas terbaru akun di entitas AWS Organizations yang spesifik atau cabang dari entitas tersebut.

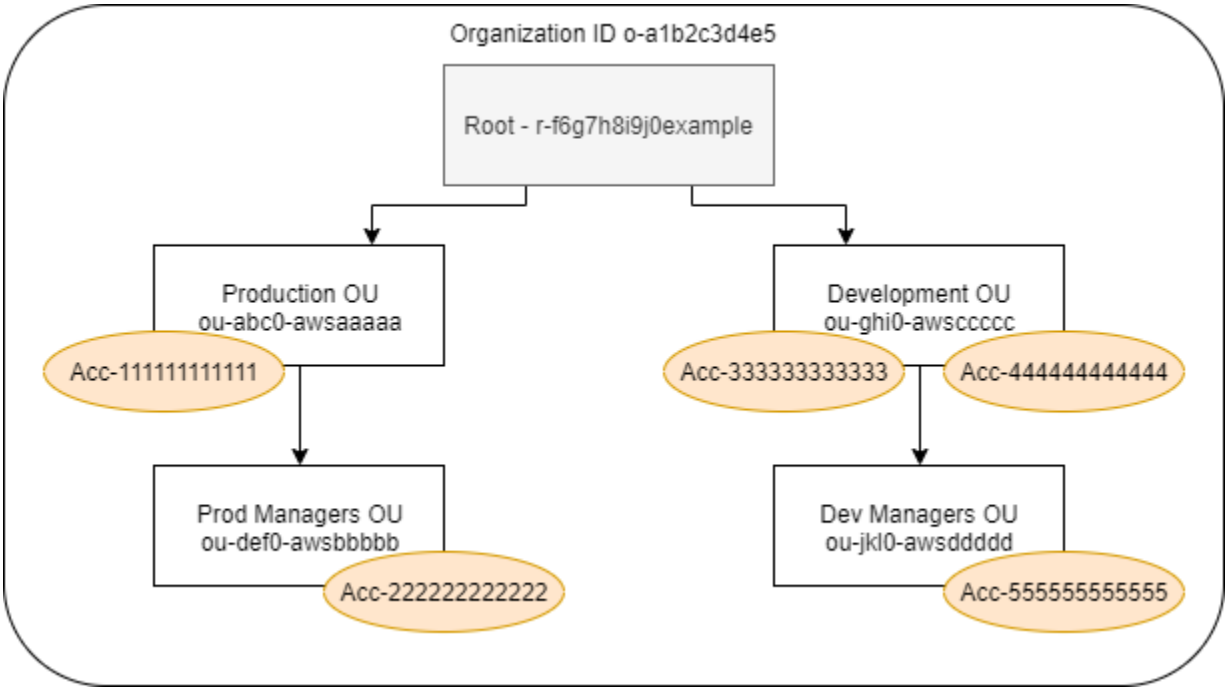
Saat membuat laporan terprogram untuk kebijakan, Anda harus menentukan AWS Organizations entitas. Laporan ini mencakup daftar layanan yang diizinkan oleh SCP tertentu. Untuk setiap layanan, ia mencakup aktivitas akun terbaru dalam entitas atau anak entitas yang diberikan izin oleh kebijakan tersebut. Untuk informasi lebih lanjut, lihat buat-laporan-akses-organisasi aws iam atau GenerateOrganizationsAccessReport.

Sebelum melihat laporan, pastikan bahwa Anda memahami persyaratan dan informasi akun manajemen, periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi. Untuk lebih detailnya, lihat the section called “Hal yang perlu diketahui tentang informasi yang terakhir diakses”.

Memahami AWS Organizations jalur entitas

Saat Anda menggunakan AWS API AWS CLI or untuk menghasilkan laporan AWS Organizations akses, Anda harus menentukan jalur entitas. Jalur adalah representasi teks dari struktur entitas AWS Organizations .

Anda dapat membangun jalur entitas menggunakan struktur Organisasi Anda yang sudah Anda pahami. Misalnya, asumsikan bahwa Anda memiliki struktur organisasi berikut di AWS Organizations.



Jalur untuk OU Manajer Pengembangan disusun dengan menggunakan ID dalam organisasi, root, dan semua OU dalam alur menuju dan mencakup OU.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsddddd/
```

Jalur masuk ke akun di Produksi OU dibangun menggunakan ID dari organisasi, root, OU, dan nomor akun.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> ID organisasi unik secara global tetapi ID OU dan root hanya unik dalam organisasinya. Ini berarti bahwa tidak ada dua organisasi yang memiliki ID organisasi yang sama. Namun, organisasi lain mungkin memiliki OU atau root dengan ID yang sama seperti milik Anda. Kami merekomendasikan agar Anda selalu menyertakan ID organisasi saat menentukan OU atau root.

Melihat informasi untuk AWS Organizations (konsol)

Anda dapat menggunakan konsol IAM untuk melihat informasi yang terakhir diakses untuk root, OU, akun, atau kebijakan Anda.

Untuk melihat informasi root (konsol)

1. Masuk ke kreden AWS Organizations sional akun manajemen Konsol Manajemen AWS menggunakan, dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di halaman Aktivitas organisasi, pilih Akar.
4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi tersebut mencakup daftar layanan yang diizinkan oleh kebijakan yang dilampirkan secara langsung pada akar. Informasi tersebut menunjukkan kepada Anda akun mana yang terakhir diakses dan kapan. Untuk detail lebih lanjut tentang penanggung jawab yang mengakses layanan, masuk sebagai administrator di akun tersebut, dan lihat informasi akses terakhir layanan IAM.
5. Pilih tab SCPs Terlampir untuk melihat daftar kebijakan kontrol layanan (SCP) yang dilampirkan ke root. IAM menunjukkan jumlah entitas target yang tertera entitas pada masing-masing kebijakan. Anda dapat menggunakan informasi ini untuk memutuskan SCP mana yang akan ditinjau.

6. Pilih nama SCP untuk melihat semua layanan yang diizinkan kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.
7. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di AWS Organizations konsol. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Untuk melihat informasi untuk OU atau akun (konsol)

1. Masuk ke kreden AWS Organizations sional akun manajemen Konsol Manajemen AWS menggunakan, dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di Aktivitas organisasi, buka lebih jauh ke struktur organisasi Anda. Kemudian pilih nama OU atau akun apa pun yang ingin Anda lihat kecuali akun manajemen.
4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi ini mencakup daftar layanan yang diizinkan oleh SCP terlampir pada OU atau akun dan semua asalnya. Informasi tersebut menunjukkan kepada Anda akun mana yang terakhir diakses dan kapan. Untuk detail lebih lanjut tentang penanggung jawab yang mengakses layanan, masuk sebagai administrator di akun tersebut, dan lihat informasi akses terakhir layanan IAM.
5. Pilih tab SCPs Terlampir untuk melihat daftar kebijakan kontrol layanan (SCP) yang dilampirkan langsung ke OU atau akun. IAM menunjukkan jumlah entitas target yang tertera entitas pada masing-masing kebijakan. Anda dapat menggunakan informasi ini untuk memutuskan SCP mana yang akan ditinjau.
6. Pilih nama SCP untuk melihat semua layanan yang diizinkan kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.
7. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di AWS Organizations konsol. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Untuk melihat informasi akun manajemen (konsol)

1. Masuk ke kreden AWS Organizations sional akun manajemen Konsol Manajemen AWS menggunakan, dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di bawah bagian Mengakses laporan, pilih Aktivitas organisasi.
3. Di halaman Aktivitas organisasi buka lebih lanjut struktur organisasi Anda dan pilih nama akun manajemen Anda.

4. Di tab Detail dan aktivitas, lihat bagian Laporan akses layanan. Informasi ini mencakup daftar semua layanan AWS . Akun manajemen tidak dibatasi oleh SCP. Informasi tersebut menunjukkan kepada Anda apakah akun terakhir mengakses layanan dan waktunya. Untuk detail lebih lanjut tentang penanggung jawab yang mengakses layanan, masuk sebagai administrator di akun tersebut, dan lihat informasi akses terakhir layanan IAM.
5. Pilih tab SCP Ter lampir untuk mengonfirmasi bahwa tidak ada SCP terlampir karena akun tersebut adalah akun manajemen.

Untuk melihat informasi untuk kebijakan (konsol)

1. Masuk ke kreden AWS Organizations sional akun manajemen Konsol Manajemen AWS menggunakan, dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di bagian bawah Mengakses laporan pilih Kebijakan kontrol layanan (SCP).
3. Di halaman Kebijakan kontrol layanan (SCP), lihat daftar kebijakan dalam organisasi Anda. IAM menunjukkan jumlah entitas target yang terlampirkan dengan masing-masing kebijakan.
4. Pilih nama SCP untuk melihat semua layanan yang diizinkan kebijakan tersebut. Untuk setiap layanan, lihat dari akun mana layanan tersebut terakhir diakses, dan kapan.
5. Pilih Edit AWS Organizations untuk melihat detail tambahan dan mengedit SCP di AWS Organizations konsol. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

Melihat informasi untuk AWS Organizations (AWS CLI)

Anda dapat menggunakan AWS CLI untuk mengambil informasi layanan yang terakhir diakses untuk AWS Organizations root, OU, akun, atau kebijakan Anda.

Untuk melihat AWS Organizations layanan informasi terakhir diakses (AWS CLI)

1. Gunakan kreden AWS Organizations tif akun manajemen Anda dengan IAM dan AWS Organizations izin yang diperlukan, dan konfirmasikan bahwa SCP diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.
2. Buat laporan. Permintaan harus menyertakan jalur AWS Organizations entitas (root, OU, atau akun) yang ingin Anda laporkan. Anda secara opsional dapat menyertakan parameter `organization-policy-id` untuk melihat laporan kebijakan tertentu. Perintah menyatakan

`job-id` yang kemudian dapat Anda gunakan di `get-organizations-access-report` perintah untuk memonitor `job-status` hingga pekerjaan selesai.

- aws iam membuat-organisasi-akses-laporan-

3. Dapatkan detail laporan menggunakan parameter `job-id` dari langkah sebelumnya.

   - aws iam dapat-organisasi-akses-laporan

   Perintah ini menghasilkan daftar layanan yang dapat diakses oleh anggota entitas. Untuk setiap layanan, perintah menyatakan tanggal dan waktu upaya terakhir anggota akun dan alur entitas akun. Ia juga menyatakan jumlah total layanan yang tersedia untuk diakses dan jumlah layanan yang tidak diakses. Jika Anda menentukan pilihan `organizations-policy-id` parameter, maka layanan yang tersedia untuk diakses adalah layanan yang diizinkan oleh kebijakan tertentu.

Melihat informasi untuk AWS Organizations (AWS API)

Anda dapat menggunakan AWS API untuk mengambil informasi layanan yang terakhir diakses untuk AWS Organizations root, OU, akun, atau kebijakan Anda.

Untuk melihat AWS Organizations layanan informasi terakhir diakses (AWS API)

1. Gunakan kreden AWS Organizations tif akun manajemen Anda dengan IAM dan AWS Organizations izin yang diperlukan, dan konfirmasikan bahwa SCP diaktifkan untuk root Anda. Untuk informasi selengkapnya, lihat Hal yang perlu diketahui tentang informasi yang terakhir diakses.
2. Buat laporan. Permintaan harus menyertakan jalur AWS Organizations entitas (root, OU, atau akun) yang ingin Anda laporkan. Anda secara opsional dapat menyertakan parameter `OrganizationsPolicyId` untuk melihat laporan kebijakan tertentu. Operasi menghasilkan `JobId` yang kemudian dapat Anda gunakan di `GetOrganizationsAccessReport` operasi untuk memonitor `JobStatus` hingga pekerjaan selesai.

   - GenerateOrganizationsAccessReport

3. Dapatkan detail laporan menggunakan parameter `JobId` dari langkah sebelumnya.

   - GetOrganizationsAccessReport

Operasi ini menyatakan daftar layanan yang dapat diakses oleh anggota entitas. Untuk setiap layanan, perintah menyatakan tanggal dan waktu upaya terakhir anggota akun dan alur entitas akun. Ia juga menyatakan jumlah total layanan yang tersedia untuk diakses, dan jumlah layanan yang tidak diakses. Jika Anda menentukan pilihan `OrganizationsPolicyId` parameter, maka layanan yang tersedia untuk diakses adalah layanan yang diizinkan oleh kebijakan tertentu.

Contoh alur perencanaan untuk menggunakan informasi yang terakhir diakses

Anda dapat menggunakan informasi yang terakhir diakses untuk membuat keputusan tentang izin yang Anda berikan kepada entitas atau AWS Organizations entitas IAM Anda. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

> **Note**
>
> Sebelum Anda melihat informasi akses untuk entitas atau kebijakan di IAM atau AWS Organizations, pastikan Anda memahami periode pelaporan, entitas yang dilaporkan, dan jenis kebijakan yang dievaluasi untuk data Anda. Untuk lebih detailnya, lihat the section called "Hal yang perlu diketahui tentang informasi yang terakhir diakses".

Sebagai administrator Anda bebas untuk menyeimbangkan ketersediaan akses dan hak istimewa terkecil untuk perusahaan Anda.

Menggunakan informasi untuk mengurangi izin bagi grup IAM

Anda dapat menggunakan informasi yang terakhir diakses untuk mengurangi izin grup IAM hanya untuk menyertakan layanan yang dibutuhkan pengguna Anda. Metode ini merupakan langkah penting dalam memberikan hak istimewa terkecil pada tingkat layanan.

Misalnya, Paulo Santos adalah administrator yang bertanggung jawab untuk menentukan izin AWS pengguna untuk Example Corp Perusahaan ini baru saja mulai menggunakan AWS, dan tim pengembangan perangkat lunak belum menentukan AWS layanan apa yang akan mereka gunakan. Paulo ingin memberikan izin kepada tim untuk hanya mengakses layanan yang mereka butuhkan, tetapi karena itu belum ditentukan, dia sementara memberikan izin penggunaan daya. Kemudian dia menggunakan informasi yang diakses terakhir untuk mengurangi izin kelompok.

Paulo membuat kebijakan terkelola bernama `ExampleDevelopment` menggunakan teks JSON berikut. Lalu ia melampirkannya ke kelompok bernama `Development` dan menambahkan semua developer ke kelompok.

> **Note**
>
> Pengguna daya Paulo mungkin membutuhkan izin `iam:CreateServiceLinkedRole` untuk menggunakan beberapa layanan dan fitur. Dia mengerti bahwa dia menambahkan izin ini memungkinkan pengguna membuat peran terkait layanan apa pun. Dia menerima risiko ini untuk pengguna dayanya.

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo memutuskan untuk menunggu selama 90 hari sebelum melihat informasi yang terakhir diakses untuk `Development` menggunakan Konsol Manajemen AWS. Dia melihat daftar layanan yang diakses anggota grup. Dia mengetahui bahwa pengguna mengakses lima layanan dalam seminggu terakhir: AWS CloudTrail, Amazon CloudWatch Logs, Amazon EC2, AWS KMS, dan Amazon S3. Mereka mengakses beberapa layanan lain ketika mereka pertama kali mengevaluasi AWS, tetapi tidak sejak itu.

Paulo memutuskan untuk mengurangi izin kebijakan untuk hanya menyertakan lima layanan tersebut dan IAM dan AWS Organizations tindakan yang diperlukan. Dia menyunting `ExampleDevelopment` menggunakan teks JSON berikut.

> **Note**
>
> Pengguna daya Paulo mungkin membutuhkan izin `iam:CreateServiceLinkedRole` untuk menggunakan beberapa layanan dan fitur. Dia mengerti bahwa dia menambahkan izin ini memungkinkan pengguna membuat peran terkait layanan apa pun. Dia menerima risiko ini untuk pengguna dayanya.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk lebih mengurangi izin, Paulo dapat melihat kegiatan akun di AWS CloudTrail Riwayat peristiwa. Di sana, ia dapat melihat informasi kegiatan terperinci yang dapat ia gunakan untuk mengurangi izin kebijakan untuk hanya mencakup tindakan dan sumber daya yang dibutuhkan oleh developer. Untuk informasi selengkapnya, lihat CloudTrail Melihat Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.

Menggunakan informasi untuk mengurangi izin bagi pengguna IAM

Anda dapat menggunakan informasi yang diakses terakhir untuk mengurangi izin bagi pengguna IAM secara individu.

Misalnya, Martha Rivera adalah administrator TI yang bertanggung jawab untuk memastikan bahwa orang-orang di perusahaannya tidak memiliki AWS izin berlebih. Sebagai bagian dari pemeriksaan keamanan berkala, dia meninjau izin semua pengguna IAM. Salah satu pengguna ini adalah developer aplikasi bernama Nikhil Jayashankar, yang sebelumnya telah memegang peran sebagai teknisi IT Karena perubahan dalam persyaratan pekerjaan, Nikhil adalah anggota baik grup `app-dev` maupun grup `security-team`. Grup `app-dev` sebagai pekerjaan barunya memberikan izin untuk beberapa layanan termasuk Amazon EC2 Amazon EBS Auto Scaling Amazon S3 Route 53, dan Elastic Transcoder. Grup `security-team` sebagai pekerjaan lama memberikan izin untuk IAM dan CloudTrail.

Sebagai administrator, Martha masuk ke konsol IAM dan memilih Pengguna, memilih nama`nikhilj`, dan kemudian memilih tab Akses Terakhir.

Martha men injau kolom Akses Terakhir dan memperhatikan bahwa Nikhil belum mengakses IAM,, Route 53 CloudTrail, Amazon Elastic Transcoder, dan sejumlah layanan lainnya. AWS Nikhil telah mengakses Amazon S3. Martha memilih S3 dari daftar layanan dan mengetahui bahwa Nikhil telah melakukan beberapa `List` tindakan Amazon S3 dalam dua minggu terakhir. Di dalam perusahaannya, Martha menegaskan bahwa Nikhil tidak memiliki kebutuhan bisnis untuk mengakses IAM dan CloudTrail lagi karena dia tidak lagi menjadi anggota tim keamanan internal.

Martha sekarang siap bertindak atas layanan tersebut dan tindakan informasi yang terakhir diakses. Namun, tidak seperti grup pada contoh sebelumnya, pengguna IAM seperti `nikhilj` mungkin tunduk pada beberapa kebijakan dan menjadi anggota dari beberapa kelompok. Martha harus berhati-hati agar tidak mengganggu akses `nikhilj` atau anggota grup lainnya. Sebagai tambahan dalam hal untuk mempelajari apa akses yang harus dimiliki Nikhil, ia harus menentukan cara dia menerima izin ini.

Martha memilih tab Izin, tempatnya melihat kebijakan mana yang terkait secara langsung ke `nikhilj` dan yang terlampir di grup. Ia membuka lebih lanjut setiap kebijakan dan melihat ringkasan kebijakan untuk mempelajari kebijakan yang memungkinkan akses ke layanan yang tidak digunakan oleh Nikhil:

- IAM — Kebijakan ter `IAMFullAccess` AWS kelola dilampirkan langsung ke `nikhilj` dan dilampirkan ke `security-team` grup.
- CloudTrail — Kebijakan `AWS CloudTrailReadOnlyAccess` AWS terkelola dilampirkan ke `security-team` grup.
- Route 53 – Kebijakan terkelola pelanggan `App-Dev-Route53` dilampirkan pada grup `app-dev`.
- Transkoder Elastis – Kebijakan terkelola pelanggan `App-Dev-ElasticTranscoder` terlampir pada grup `app-dev`.

Martha memutuskan untuk menghapus kebijakan `IAMFullAccess` AWS terkelola yang dilampirkan langsung ke`nikhilj`. Dia juga menghapus keanggotaan Nikhil ke grup `security-team`. Kedua tindakan ini menghapus akses yang tidak perlu ke IAM dan CloudTrail.

Izin Nikhil untuk mengakses Route 53 dan Transkoder Elastis diberikan oleh grup `app-dev`. Meskipun Nikhil tidak menggunakan layanan tersebut, anggota lain kelompok tersebut mungkin saja demikian. Martha meninjau informasi yang terakhir diakses untuk grup `app-dev` dan mempelajari bahwa beberapa anggota mengakses Route 53 dan Amazon S3 baru-baru ini. Namun, tidak ada anggota kelompok yang telah mengakses Transkoder Elastis tahun lalu. Dia menghapus kebijakan pengelolaan pelanggan `App-Dev-ElasticTranscoder` dari grup.

Martha kemudian meninjau informasi yang terakhir diakses untuk `App-Dev-ElasticTranscoder` kebijakan terkelola pelanggan. Dia belajar bahwa kebijakan tersebut tidak terkait dengan identitas IAM lainnya. Ia menginvestigasi dalam perusahaannya untuk memastikan bahwa kebijakan tersebut tidak diperlukan di waktu yang akan datang, kemudian ia menghapusnya.

Menggunakan informasi sebelum menghapus sumber daya IAM

Anda dapat menggunakan informasi yang terakhir diakses sebelum Anda menghapus sumber daya IAM untuk memastikan bahwa sudah sekian lama sejak seseorang terakhir menggunakan sumber daya tersebut. Ini berlaku untuk pengguna, grup, peran, dan kebijakan. Untuk mempelajari lebih lanjut tentang tindakan ini, lihat topik berikut:

- Pengguna IAM — Mengh apus atau menonaktifkan pengguna IAM
- Grup — Menghapus grup IAM
- Per an — Hapus peran atau profil instance
- Kebijakan — Hapus kebijakan IAM (ini juga melepaskan kebijakan dari identitas)

Menggunakan informasi sebelum menyunting kebijakan IAM

Anda dapat meninjau informasi terakhir yang diakses untuk identitas IAM (pengguna, grup, atau peran), atau untuk kebijakan IAM sebelum menyunting kebijakan yang memengaruhi sumber daya tersebut. Ini penting karena Anda tidak ingin menghapus akses untuk orang yang menggunakannya.

Misalnya, Arnav Desai adalah pengembang dan AWS administrator untuk Example Corp. Ketika timnya mulai menggunakan AWS, mereka memberi semua pengembang akses pengguna daya yang memungkinkan mereka akses penuh ke semua layanan kecuali IAM dan. AWS Organizations Sebagai langkah pertama untuk memberikan hak istimewa paling rendah, Arnav ingin menggunakan AWS CLI untuk meninjau kebijakan yang dikelola dalam akunnya.

Untuk melakukannya, Arnav terlebih dahulu mencantumkan kebijakan izin yang dikelola pelanggan di akunnya yang terlampir ke suatu identitas, menggunakan perintah berikut:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

Dari respons tersebut, dia mengambil ARN untuk setiap kebijakan. Arnav membuat laporan untuk informasi yang terakhir diakses bagi setiap kebijakan dengan menggunakan perintah berikut.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dari respons tersebut, dia mengambil ID dari laporan yang dihasilkan dari bidang `JobId`. Arnav kemudian melakukan jajak pendapat perintah berikut sampai bidang `JobStatus` menghasilkan nilai `COMPLETED` atau `FAILED`. Jika tugas gagal, ia akan menangkap kesalahan.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Ketika pekerjaan memiliki status`COMPLETED`, Arnav mengurai isi array. JSON-formatted `ServicesLastAccessed`

```
 "ServicesLastAccessed": [
        {
            "TotalAuthenticatedEntities": 1,
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
    ]
```

Dari informasi ini, Arnav mengetahui bahwa `ExamplePolicy1` kebijakan tersebut memungkinkan akses ke tiga layanan, Amazon DynamoDB, Amazon S3, dan Amazon EC2. Pengguna IAM bernama `IAMExampleUser` terakhir mencoba mengakses DynamoDB pada 1 November, dan seseorang menggunakan `IAMExampleRole` peran tersebut untuk mencoba mengakses Amazon S3 pada 25 Agustus. Ada pula dua entitas lagi yang mencoba mengakses Amazon S3 tahun lalu. Namun, tidak ada yang telah mencoba mengakses Amazon EC2 tahun lalu.

Ini berarti bahwa Arnav dapat dengan aman menghapus tindakan Amazon EC2 dari kebijakan. Arnav ingin meninjau berkas JSON saat ini untuk kebijakan tersebut. Pertama, ia harus menentukan nomor versi kebijakan menggunakan perintah berikut.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dari respons tersebut, Arnav mengumpulkan nomor versi default saat ini dari himpunan `Versions`. Kemudian dia menggunakan nomor versi tersebut (`v2`) meminta berkas kebijakan JSON dengan menggunakan perintah berikut.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav menyimpan dokumen kebijakan JSON yang dikembalikan di `Document` bidang himpunan `PolicyVersion`. Dalam dokumen kebijakan, Arnav mencari tindakan yang dalam namespace `ec2`. Jika tidak ada tindakan dari namespace lain yang tersisa dalam kebijakan, maka dia memisahkan kebijakan dari identitas yang terdampak (pengguna, grup, dan peran). Kemudian, dia menghapus kebijakan tersebut. Dalam hal ini, kebijakan tersebut mencakup layanan Amazon DynamoDB dan Amazon S3. Jadi Arnav menghapus tindakan Amazon EC2 dari dokumen dan menyimpan perubahannya. Kemudian, ia menggunakan perintah berikut untuk memperbarui kebijakan dengan menggunakan versi baru berkas tersebut dan menetapkan versi tersebut sebagai versi kebijakan standar.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

`ExamplePolicy1` kebijakan ini diperbarui untuk menghapus akses ke layanan Amazon EC2 yang tidak perlu.

Skenario IAM lainnya

Informasi tentang ketika sumber daya IAM (pengguna, grup, peran, atau kebijakan) yang terakhir kali diupayakan untuk mengakses layanan dapat membantu Anda setelah menyelesaikan tugas berikut:

- Kebijakan – Menyunting kebijakan yang dikelola pelanggan atau kebijakan selaras yang sudah ada untuk menghapus izin
- Kebijakan – Mengonversi kebijakan selaras ke kebijakan terkelola dan kemudian menghapusnya
- Kebijakan – Menambahkan penolakan secara eksplisit ke kebijakan yang sudah ada
- Policies – Melepas kebijakan terkelola dari suatu identitas (pengguna, grup, atau peran)
- Entitas – Atur batas izin untuk mengendalikan perizinan maksimum yang dapat dimiliki oleh entitas (pengguna atau peran)

- Kelompok – Menghapus pengguna dari grup

Menggunakan informasi untuk memperbaiki izin unit organisasi.

Anda dapat menggunakan informasi yang diakses terakhir untuk memperbaiki izin unit organisasi (OU) di AWS Organizations.

Misalnya, John Stiles adalah seorang AWS Organizations administrator. Dia bertanggung jawab untuk memastikan bahwa orang-orang di perusahaan Akun AWS tidak memiliki izin berlebih. Sebagai bagian dari audit keamanan berkala, ia meninjau izin dari organisasinya. OU `Development`-nya berisikan akun yang sering digunakan untuk menguji layanan AWS baru. John memutuskan untuk secara berkala meninjau laporan bagi layanan yang belum diakses dalam lebih dari 180 hari. Kemudian, ia menghapus izin bagi anggota OU untuk mengakses layanan tersebut.

John masuk ke konsol IAM menggunakan kredensial akun manajemennya. Di konsol IAM, ia menemukan AWS Organizations data untuk OU`Development`. Dia meninjau tabel laporan akses Layanan dan melihat dua AWS layanan yang belum diakses dalam lebih dari periode pilihannya 180 hari. Dia ingat menambahkan izin bagi tim pengembangan untuk mengakses Amazon Lex dan AWS Database Migration Service. John menghubungi tim pengembangan dan mengonfirmasi bahwa mereka tidak lagi memiliki kepentingan bisnis untuk menguji layanan ini.

John sekarang siap bertindak atas informasi yang terakhir diakses. Dia memilih Edit di AWS Organizations dan diingatkan bahwa SCP melekat pada beberapa entitas. Dia memilih Lanjutkan. Pada AWS Organizations tahun, ia meninjau target untuk mempelajari AWS Organizations entitas mana yang dilampirkan SCP. Semua entitas berada dalam OU `Development`.

John memutuskan untuk menolak akses ke Amazon Lex dan AWS Database Migration Service tindakan di `NewServiceTest` SCP. Tindakan ini menghapus akses yang tidak perlu ke layanan.

Tindakan IAM terakhir mengakses layanan dan tindakan informasi

Tabel berikut mencantumkan AWS layanan untuk tindakan IAM yang terakhir diakses informasi ditampilkan. Untuk daftar tindakan di setiap layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

AWS menyediakan informasi tindakan yang terakhir diakses dalam format JSON untuk merampingkan otomatisasi alur kerja manajemen kebijakan. Dengan informasi referensi layanan, Anda dapat mengakses informasi tindakan yang terakhir diakses di Layanan AWS seberang file yang dapat dibaca mesin. Untuk informasi selengkapnya, lihat Layanan AWS Informasi yang disederhanakan untuk akses terprogram di Referensi Otorisasi Layanan.

Layanan	Awalan layanan
AWS Identity and Access Management dan Access Analyzer	akses-analyzer
AWS Account Management	akun
AWS Certificate Manager	acm
Alur Kerja Terkelola Amazon untuk Apache Airflow	aliran udara
AWS Amplify	amplify
AWS Amplify Pembuat UI	amplifyuibuilder
Amazon AppIntegrations	integrasi aplikasi
AWS AppConfig	appconfig
Amazon AppFlow	Appflow
AWS Profiler Biaya Aplikasi	aplikasi-biaya-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi
AWS App Mesh	appmesh
WorkSpaces Aplikasi Amazon	appstream
AWS AppSync	appsync
Layanan Terkelola Amazon untuk Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	Pengelola Audit
AWS Auto Scaling	penskalaan otomatis
AWS Marketplace	aws-marketplace

Layanan	Awalan layanan
AWS Backup	pencadangan
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	anggaran
AWS Cloud9	awan9
AWS CloudFormation	pembentukan awan
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	kodeartefak
AWS CodeDeploy	penyebaran kode
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Peninjau Amazon	pengulas codeguru
AWS CodePipeline	codepipeline
AWS CodeStar	bintang kode
AWS CodeStar Pemberitahuan	pemberitahuan bintang kode
Identitas Amazon Cognito	kognito-identitas

Layanan	Awalan layanan
Kumpulan pengguna Amazon Cognito	cognito-idp
Sinkronisasi Amazon Cognito	sinkronisasi kognito
Amazon Comprehend Medical	memahami-medis
AWS Compute Optimizer	pengoptimal komputasi
AWS Config	config
Pelanggan Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
DevOpsGuru Amazon	devops-guru
AWS Direct Connect	DirectConnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Cluster Elastis Amazon DocumentDB	docdb-elastis
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs

Layanan	Awalan layanan
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Publik	ecr-publik
Layanan Kontainer Elastis Amazon	ecs
Layanan Amazon Elastic Kubernetes	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	tangkai beanstalk
Sistem File Elastis Amazon	sistem file elastis
Elastic Load Balancing	elasticlo adbalancing
Amazon EMR di EKS (Wadah EMR)	kontainer emr
Amazon EMR Tanpa Server	emr-tanpa server
OpenSearch Layanan Amazon	es
Amazon EventBridge	peristiwa
Amazon FinSpace	ruang finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	detektor penipuan
Amazon FSx	fsx
GameLift Peladen Amazon	gamelift

Layanan	Awalan layanan
Amazon Location Service	geo
Amazon Glacier	gletser
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	stasiun tanah
Amazon GuardDuty	tugas jaga
AWS HealthLake	healthlake
Honeycode Amazon	kode madu
AWS Identity and Access Management	iam
AWS Toko Identitas	toko identitas
EC2 Image Builder	imagebuilder
Amazon Inspector Klasik	inspektur
Amazon Inspector	inspektor2
AWS IoT	iot
AWS IoT Analytics	iotanalitik
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	pembuat iottwinmaker

Layanan	Awalan layanan
AWS IoT Wireless	iotwireless
Layanan Video Interaktif Amazon	ivs
Obrolan Layanan Video Interaktif Amazon	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalitik
AWS Key Management Service	km
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Manajer Langganan Linux	lisensi-manajer-linux-langganan
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect

Layanan	Awalan layanan
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Rekomendasi Strategi Migrasi Hub	Migration hub-strategi
Amazon Pinpoint	penargetan seluler
Amazon MQ	mq
AWS Network Manager	manajer jaringan
Studio Amazon Nimble	gesit
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	pos terdepan
AWS Organizations	organisasi
AWS Performance Insights	pi
EventBridgePipa Amazon	pipa
Amazon Polly	polly
Profil Pelanggan Pelanggan Amazon Connect	profile

Layanan	Awalan layanan
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Tempat Sampah Daur Ulang	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API Data Pergeseran Merah Amazon	data pergesera n merah
AWS Migration Hub Refactor Spaces	ruang refaktor
Amazon Rekognition	rekognisi
AWS Resilience Hub	resiliencehub
Penjelajah Sumber Daya AWS	sumber daya-penjelajah-2
AWS Resource Groups	kelompok sumber daya
AWS RoboMaker	robomaker
AWS Identity and Access Management Peran Di Mana Saja	peranandi mana saja
Rute Amazon 53	route53
Amazon Route 53 Kontrol Pemulihan	route53-recovery-control-config
Kesiapan Pemulihan Amazon Route 53	route53-p emulihan-kesiapan
Amazon Route 53 Resolver	route53resolver

Layanan	Awalan layanan
AWS CloudWatchRUM	rum
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 on Outposts	pos terdepan s3
Kemampuan geospasial Amazon SageMaker AI	sagemaker-geospasial
Savings Plans	savingsplans
EventBridgeSkema Amazon	skema
Amazon SimpleDB	sdb
AWS Secrets Manager	manajer rahasia
AWS Security Hub CSPM	securityhub
Amazon Security Lake	Securitylake
AWS Serverless Application Repository	repo tanpa server
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Layanan Email Amazon Sederhana	ses
AWS Shield	melindungi
AWS Signer	penandatangan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara

Layanan	Awalan layanan
AWS Snowball Edge	bola salju
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
Manajer Insiden AWS Systems Manager	insiden ssm
Manajer Sistem AWS untuk SAP	ssm-sap
AWS Step Functions	negara
AWS Security Token Service	sts
Layanan Alur Kerja Sederhana Amazon	swf
Amazon CloudWatch Synthetics	sintetis
AWS Resource Groups Tagging API	tanda
Amazon Texttract	textract
Amazon Timestream	aliran waktu
AWS Pembangun Jaringan Telekomunikasi	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
Connect ID Suara Pelanggan	voiceid
Kisi VPC Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected

Layanan	Awalan layanan
Kebijaksanaan Pelanggan Amazon Connect	kebijaksanaan
Amazon WorkLink	tautan kerja
Amazon WorkSpaces	ruang kerja
AWS X-Ray	xray

Tindakan untuk tindakan informasi yang terakhir diakses

Tabel berikut mencantumkan tindakan untuk tindakan yang terakhir diakses informasi yang tersedia.

> **Important**
>
> `iam:UpdateAccountName`Tindakan tersebut akan dihentikan pada 22 April 2026. Setelah 22 April 2026, hanya `account:PutAccountName` izin yang akan mengontrol akses pembaruan nama akun. Kami sangat menyarankan agar Anda memperbarui kebijakan kontrol layanan (SCP) apa pun yang mengontrol pembaruan nama akun untuk menggunakan `account:PutAccountName` izin.

Awalan layanan	Tindakan
akses-analyzer	akses-analyzer: ApplyArchiveRule
	akses-analyzer: CancelPolicyGeneration
	akses-analyzer: CheckAccessNotGranted
	akses-analyzer: CheckNoNewAccess
	akses-analyzer: CheckNoPublicAccess
	akses-analyzer: CreateAccessPreview
	akses-analyzer: CreateAnalyzer
	akses-analyzer: CreateArchiveRule

Awalan layanan	Tindakan
	akses-analyzer: DeleteAnalyzer
	akses-analyzer: DeleteArchiveRule
	akses-analyzer: GenerateFindingRecommendation
	akses-analyzer: GetAccessPreview
	akses-analyzer: GetAnalyzedResource
	akses-analyzer: GetAnalyzer
	akses-analyzer: GetArchiveRule
	akses-analyzer: GetFinding
	akses-analyzer: GetFindingRecommendation
	akses-analyzer: GetFindingsStatistics
	akses-analyzer: GetGeneratedPolicy
	akses-analyzer: ListAccessPreviewFindings
	akses-analyzer: ListAccessPreviews
	akses-analyzer: ListAnalyzedResources
	akses-analyzer: ListAnalyzers
	akses-analyzer: ListArchiveRules
	akses-analyzer: ListFindings
	akses-analyzer: ListPolicyGenerations
	akses-analyzer: StartPolicyGeneration
	akses-analyzer: StartResourceScan
	akses-analyzer: UpdateAnalyzer

Awalan layanan	Tindakan
	akses-analyzer: UpdateArchiveRule akses-analyzer: UpdateFindings akses-analyzer: ValidatePolicy
akun	akun: AcceptPrimaryEmailUpdate akun: DeleteAlternateContact akun: DisableRegion akun: EnableRegion akun: GetAccountInformation akun: GetAlternateContact akun: GetContactInformation akun: GetGovCloudAccountInformation akun: GetPrimaryEmail akun: GetRegionOptStatus akun: ListRegions akun: PutAccountName akun: PutAlternateContact akun: PutContactInformation akun: StartPrimaryEmailUpdate

Awalan layanan	Tindakan
acm	ACM: CreateAcmeEndpoint
	ACM: DeleteCertificate
	ACM: DescribeCertificate
	ACM: ExportCertificate
	ACM: GetAccountConfiguration
	ACM: GetCertificate
	ACM: ImportCertificate
	ACM: ListAcmeEndpoints
	ACM: ListCertificates
	ACM: PutAccountConfiguration
	ACM: RenewCertificate
	ACM: RequestCertificate
	ACM: ResendValidationEmail
	ACM: SearchCertificates
	ACM: UpdateCertificateOptions

Awalan layanan	Tindakan
aliran udara	aliran udara: CreateCliToken aliran udara: CreateEnvironment aliran udara: CreateWebLoginToken aliran udara: DeleteEnvironment aliran udara: GetEnvironment aliran udara: ListEnvironments aliran udara: PublishMetrics aliran udara: UpdateEnvironment

Awalan layanan	Tindakan
amplify	memperkuat: CreateApp
	memperkuat: CreateBackendEnvironment
	memperkuat: CreateBranch
	memperkuat: CreateDeployment
	memperkuat: CreateDomainAssociation
	memperkuat: CreateWebHook
	memperkuat: DeleteApp
	memperkuat: DeleteBackendEnvironment
	memperkuat: DeleteBranch
	memperkuat: DeleteDomainAssociation
	memperkuat: DeleteJob
	memperkuat: DeleteWebHook
	memperkuat: GenerateAccessLogs
	memperkuat: GetApp
	memperkuat: GetArtifactUrl
	memperkuat: GetBackendEnvironment
	memperkuat: GetBranch
	memperkuat: GetDomainAssociation
	memperkuat: GetJob
	memperkuat: GetWebHook
	memperkuat: ListApps

Awalan layanan	Tindakan
	memperkuat: ListArtifacts
	memperkuat: ListBackendEnvironments
	memperkuat: ListBranches
	memperkuat: ListDomainAssociations
	memperkuat: ListJobs
	memperkuat: ListWebHooks
	memperkuat: StartDeployment
	memperkuat: StartJob
	memperkuat: StopJob
	memperkuat: UpdateApp
	memperkuat: UpdateBranch
	memperkuat: UpdateDomainAssociation
	memperkuat: UpdateWebHook

Awalan layanan	Tindakan
amplifyuibuilder	amplifyuibuilder: CreateComponent amplifyuibuilder: CreateForm amplifyuibuilder: CreateTheme amplifyuibuilder: DeleteComponent amplifyuibuilder: DeleteForm amplifyuibuilder: DeleteTheme amplifyuibuilder: ExportComponents amplifyuibuilder: ExportThemes amplifyuibuilder: GetCodegenJob amplifyuibuilder: ListCodegenJobs amplifyuibuilder: ListComponents amplifyuibuilder: ListForms amplifyuibuilder: ListThemes amplifyuibuilder: ResetMetadataFlag amplifyuibuilder: StartCodegenJob amplifyuibuilder: UpdateComponent amplifyuibuilder: UpdateForm amplifyuibuilder: UpdateTheme

Awalan layanan	Tindakan
integrasi aplikasi	integrasi aplikasi: CreateApplication
	integrasi aplikasi: CreateDataIntegration
	integrasi aplikasi: CreateDataIntegrationAssociation
	integrasi aplikasi: CreateEventIntegration
	integrasi aplikasi: DeleteApplication
	integrasi aplikasi: DeleteDataIntegration
	integrasi aplikasi: DeleteEventIntegration
	integrasi aplikasi: GetApplication
	integrasi aplikasi: GetDataIntegration
	integrasi aplikasi: GetEventIntegration
	integrasi aplikasi: ListApplicationAssociations
	integrasi aplikasi: ListApplications
	integrasi aplikasi: ListDataIntegrationAssociations
	integrasi aplikasi: ListDataIntegrations
	integrasi aplikasi: ListEventIntegrationAssociations
	integrasi aplikasi: ListEventIntegrations
	integrasi aplikasi: UpdateApplication
	integrasi aplikasi: UpdateDataIntegration
	integrasi aplikasi: UpdateDataIntegrationAssociation
	integrasi aplikasi: UpdateEventIntegration

Awalan layanan	Tindakan
appconfig	appconfig: CreateApplication appconfig: CreateConfigurationProfile appconfig: CreateDeploymentStrategy appconfig: CreateEnvironment appconfig: CreateExtension appconfig: CreateExtensionAssociation appconfig: CreateHostedConfigurationVersion appconfig: DeleteApplication appconfig: DeleteConfigurationProfile appconfig: DeleteDeploymentStrategy appconfig: DeleteEnvironment appconfig: DeleteExtension appconfig: DeleteExtensionAssociation appconfig: DeleteHostedConfigurationVersion appconfig: GetAccountSettings appconfig: GetApplication appconfig: GetConfiguration appconfig: GetConfigurationProfile appconfig: GetDeployment appconfig: GetDeploymentStrategy appconfig: GetEnvironment

Awalan layanan	Tindakan
	appconfig: GetExtension appconfig: GetExtensionAssociation appconfig: GetHostedConfigurationVersion appconfig: ListApplications appconfig: ListConfigurationProfiles appconfig: ListDeploymentStrategies appconfig: ListDeployments appconfig: ListEnvironments appconfig: ListExtensionAssociations appconfig: ListExtensions appconfig: ListHostedConfigurationVersions appconfig: StartDeployment appconfig: StopDeployment appconfig: UpdateAccountSettings appconfig: UpdateApplication appconfig: UpdateConfigurationProfile appconfig: UpdateDeploymentStrategy appconfig: UpdateEnvironment appconfig: UpdateExtension appconfig: UpdateExtensionAssociation appconfig: ValidateConfiguration

Awalan layanan	Tindakan
Appflow	Appflow: CancelFlowExecutions
	Appflow: CreateConnectorProfile
	Appflow: CreateFlow
	Appflow: DeleteConnectorProfile
	Appflow: DeleteFlow
	Appflow: DescribeConnector
	Appflow: DescribeConnectorEntity
	Appflow: DescribeConnectorProfiles
	Appflow: DescribeConnectors
	Appflow: DescribeFlow
	Appflow: DescribeFlowExecutionRecords
	Appflow: ListConnectorEntities
	Appflow: ListConnectors
	Appflow: ListFlows
	Appflow: RegisterConnector
	Appflow: ResetConnectorMetadataCache
	Appflow: StartFlow
	Appflow: StopFlow
	Appflow: UnRegisterConnector
	Appflow: UpdateConnectorProfile
	Appflow: UpdateConnectorRegistration

Awalan layanan	Tindakan
	Appflow: UpdateFlow

Awalan layanan	Tindakan
wawasan aplikasi	wawasan aplikasi: AddWorkload wawasan aplikasi: CreateApplication wawasan aplikasi: CreateComponent wawasan aplikasi: CreateLogPattern wawasan aplikasi: DeleteApplication wawasan aplikasi: DeleteComponent wawasan aplikasi: DeleteLogPattern wawasan aplikasi: DescribeApplication wawasan aplikasi: DescribeComponent wawasan aplikasi: DescribeComponentConfiguration wawasan aplikasi: DescribeComponentConfigurationRecommendation wawasan aplikasi: DescribeLogPattern wawasan aplikasi: DescribeObservation wawasan aplikasi: DescribeProblem wawasan aplikasi: DescribeProblemObservations wawasan aplikasi: DescribeWorkload wawasan aplikasi: ListApplications wawasan aplikasi: ListComponents wawasan aplikasi: ListConfigurationHistory wawasan aplikasi: ListLogPatternSets

Awalan layanan	Tindakan
	wawasan aplikasi: ListLogPatterns wawasan aplikasi: ListProblems wawasan aplikasi: ListWorkloads wawasan aplikasi: RemoveWorkload wawasan aplikasi: UpdateApplication wawasan aplikasi: UpdateComponent wawasan aplikasi: UpdateComponentConfiguration wawasan aplikasi: UpdateLogPattern wawasan aplikasi: UpdateWorkload

Awalan layanan	Tindakan
appmesh	appmesh: CreateGatewayRoute
	appmesh: CreateMesh
	appmesh: CreateRoute
	appmesh: CreateVirtualGateway
	appmesh: CreateVirtualNode
	appmesh: CreateVirtualRouter
	appmesh: CreateVirtualService
	appmesh: DeleteGatewayRoute
	appmesh: DeleteMesh
	appmesh: DeleteRoute
	appmesh: DeleteVirtualGateway
	appmesh: DeleteVirtualNode
	appmesh: DeleteVirtualRouter
	appmesh: DeleteVirtualService
	appmesh: DescribeGatewayRoute
	appmesh: DescribeMesh
	appmesh: DescribeRoute
	appmesh: DescribeVirtualGateway
	appmesh: DescribeVirtualNode
	appmesh: DescribeVirtualRouter
	appmesh: DescribeVirtualService

Awalan layanan	Tindakan
	appmesh: ListGatewayRoutes
	appmesh: ListMeshes
	appmesh: ListRoutes
	appmesh: ListVirtualGateways
	appmesh: ListVirtualNodes
	appmesh: ListVirtualRouters
	appmesh: ListVirtualServices
	appmesh: StreamAggregatedResources
	appmesh: UpdateGatewayRoute
	appmesh: UpdateMesh
	appmesh: UpdateRoute
	appmesh: UpdateVirtualGateway
	appmesh: UpdateVirtualNode
	appmesh: UpdateVirtualRouter
	appmesh: UpdateVirtualService

Awalan layanan	Tindakan
appstream	appstream: AssociateAppBlockBuilderAppBlock appstream: AssociateApplicationFleet appstream: AssociateApplicationToEntitlement appstream: AssociateFleet appstream: AssociateSoftwareToImageBuilder appstream: BatchAssociateUserStack appstream: BatchDisassociateUserStack appstream: CopyImage appstream: CreateAppBlock appstream: CreateAppBlockBuilder appstream: CreateAppBlockBuilderStreamingURL appstream: CreateApplication appstream: CreateDirectoryConfig appstream: CreateEntitlement appstream: CreateFleet appstream: CreateImageBuilder appstream: CreateImageBuilderStreamingURL appstream: CreateStack appstream: URL CreateStreaming appstream: CreateThemeForStack appstream: CreateUpdatedImage

Awalan layanan	Tindakan
	appstream: CreateUsageReportSubscription appstream: CreateUser appstream: DeleteAppBlock appstream: DeleteAppBlockBuilder appstream: DeleteApplication appstream: DeleteDirectoryConfig appstream: DeleteEntitlement appstream: DeleteFleet appstream: DeleteImage appstream: DeleteImageBuilder appstream: DeleteImagePermissions appstream: DeleteStack appstream: DeleteThemeForStack appstream: DeleteUsageReportSubscription appstream: DeleteUser appstream: DescribeAppBlockBuilderAppBlockAssociations appstream: DescribeAppBlockBuilders appstream: DescribeAppBlocks appstream: DescribeAppLicenseUsage appstream: DescribeApplicationFleetAssociations appstream: DescribeApplications

Awalan layanan	Tindakan
	appstream: DescribeDirectoryConfigs appstream: DescribeEntitlements appstream: DescribeFleets appstream: DescribeImageBuilders appstream: DescribeImagePermissions appstream: DescribeImages appstream: DescribeSessions appstream: DescribeStacks appstream: DescribeThemeForStack appstream: DescribeUsageReportSubscriptions appstream: DescribeUserStackAssociations appstream: DescribeUsers appstream: DisableUser appstream: DisassociateAppBlockBuilderAppBlock appstream: DisassociateApplicationFleet appstream: DisassociateApplicationFromEntitlement appstream: DisassociateFleet appstream: DisassociateSoftwareFromImageBuilder appstream: EnableUser appstream: ExpireSession appstream: GetExportImageTask

Awalan layanan	Tindakan
	appstream: ListAssociatedFleets appstream: ListAssociatedStacks appstream: ListEntitledApplications appstream: ListExportImageTasks appstream: StartAppBlockBuilder appstream: StartFleet appstream: StartImageBuilder appstream: StartSoftwareDeploymentToImageBuilder appstream: StopAppBlockBuilder appstream: StopFleet appstream: StopImageBuilder appstream: UpdateAppBlockBuilder appstream: UpdateApplication appstream: UpdateDirectoryConfig appstream: UpdateEntitlement appstream: UpdateFleet appstream: UpdateImagePermissions appstream: UpdateStack appstream: UpdateThemeForStack

Awalan layanan	Tindakan
appsync	appsync: AssociateApi
	appsync: AssociateMergedGraphqlApi
	appsync: AssociateSourceGraphqlApi
	appsync: CreateApi
	appsync: CreateApiCache
	appsync: CreateApiKey
	appsync: CreateChannelNamespace
	appsync: CreateDataSource
	appsync: CreateDomainName
	appsync: CreateFunction
	appsync: CreateGraphqlApi
	appsync: CreateResolver
	appsync: CreateType
	appsync: DeleteApi
	appsync: DeleteApiCache
	appsync: DeleteApiKey
	appsync: DeleteChannelNamespace
	appsync: DeleteDataSource
	appsync: DeleteDomainName
	appsync: DeleteFunction
	appsync: DeleteGraphqlApi

Awalan layanan	Tindakan
	appsync: DeleteResolver
	appsync: DeleteType
	appsync: DisassociateApi
	appsync: DisassociateMergedGraphqlApi
	appsync: DisassociateSourceGraphqlApi
	appsync: EvaluateCode
	appsync: EvaluateMappingTemplate
	appsync: FlushApiCache
	appsync: GetApi
	appsync: GetApiAssociation
	appsync: GetApiCache
	appsync: GetChannelNamespace
	appsync: GetDataSource
	appsync: GetDataSourceIntrospection
	appsync: GetDomainName
	appsync: GetFunction
	appsync: GetGraphqlApi
	appsync: GetGraphqlApiEnvironmentVariables
	appsync: GetIntrospectionSchema
	appsync: GetResolver
	appsync: GetSchemaCreationStatus

Awalan layanan	Tindakan
	appsync: GetSourceApiAssociation
	appsync: GetType
	appsync: ListApiKeys
	appsync: ListApis
	appsync: ListChannelNamespaces
	appsync: ListDataSources
	appsync: ListDomainNames
	appsync: ListFunctions
	appsync: ListGraphqlApis
	appsync: ListResolvers
	appsync: ListResolversByFunction
	appsync: ListSourceApiAssociations
	appsync: ListTypes
	appsync: ListTypesByAssociation
	appsync: PutGraphqlApiEnvironmentVariables
	appsync: StartDataSourceIntrospection
	appsync: StartSchemaCreation
	appsync: StartSchemaMerge
	appsync: UpdateApi
	appsync: UpdateApiCache
	appsync: UpdateApiKey

Awalan layanan	Tindakan
	appsync: UpdateChannelNamespace
	appsync: UpdateDataSource
	appsync: UpdateDomainName
	appsync: UpdateFunction
	appsync: UpdateGraphqlApi
	appsync: UpdateResolver
	appsync: UpdateSourceApiAssociation
	appsync: UpdateType

Awalan layanan	Tindakan
aps	aps: CreateAlertManagerDefinition aps: CreateAnomalyDetector aps: CreateLoggingConfiguration aps: CreateQueryLoggingConfiguration aps: CreateRuleGroupsNamespace aps: CreateWorkspace aps: DeleteAlertManagerDefinition aps: DeleteAnomalyDetector aps: DeleteLoggingConfiguration aps: DeleteQueryLoggingConfiguration aps: DeleteResourcePolicy aps: DeleteRuleGroupsNamespace aps: DeleteScraper aps: DeleteScraperLoggingConfiguration aps: DeleteWorkspace aps: DescribeAlertManagerDefinition aps: DescribeAnomalyDetector aps: DescribeLoggingConfiguration aps: DescribeQueryLoggingConfiguration aps: DescribeResourcePolicy aps: DescribeRuleGroupsNamespace

Awalan layanan	Tindakan
	aps: DescribeScraper aps: DescribeScraperLoggingConfiguration aps: DescribeWorkspace aps: DescribeWorkspaceConfiguration aps: GetDefaultScraperConfiguration aps: ListAnomalyDetectors aps: ListRuleGroupsNamespaces aps: ListScrapers aps: ListWorkspaces aps: PutAlertManagerDefinition aps: PutAnomalyDetector aps: PutResourcePolicy aps: PutRuleGroupsNamespace aps: UpdateLoggingConfiguration aps: UpdateQueryLoggingConfiguration aps: UpdateScraper aps: UpdateScraperLoggingConfiguration aps: UpdateWorkspaceAlias aps: UpdateWorkspaceConfiguration

Awalan layanan	Tindakan
athena	Athena: BatchGetNamedQuery
	Athena: BatchGetPreparedStatement
	Athena: BatchGetQueryExecution
	Athena: CancelCapacityReservation
	Athena: CreateCapacityReservation
	Athena: CreateDataCatalog
	Athena: CreateNamedQuery
	Athena: CreateNotebook
	Athena: CreatePreparedStatement
	Athena: CreatePresignedNotebookUrl
	Athena: CreateWorkGroup
	Athena: DeleteCapacityReservation
	Athena: DeleteDataCatalog
	Athena: DeleteNamedQuery
	Athena: DeleteNotebook
	Athena: DeletePreparedStatement
	Athena: DeleteWorkGroup
	Athena: ExportNotebook
	Athena: GetCalculationExecution
	Athena: GetCalculationExecutionCode
	Athena: GetCalculationExecutionStatus

Awalan layanan	Tindakan
	Athena: GetCapacityAssignmentConfiguration
	Athena: GetCapacityReservation
	Athena: GetDataCatalog
	Athena: GetDatabase
	Athena: GetNamedQuery
	Athena: GetNotebookMetadata
	Athena: GetPreparedStatement
	Athena: GetQueryExecution
	Athena: GetQueryResults
	Athena: GetQueryResultsStream
	Athena: GetQueryRuntimeStatistics
	Athena: GetResourceDashboard
	Athena: GetSession
	Athena: GetSessionEndpoint
	Athena: GetSessionStatus
	Athena: GetTableMetadata
	Athena: GetWorkGroup
	Athena: ImportNotebook
	Athena: ListApplicationDPUSizes
	Athena: ListCalculationExecutions
	Athena: ListCapacityReservations

Awalan layanan	Tindakan
	Athena: ListDataCatalogs
	Athena: ListDatabases
	Athena: ListEngineVersions
	Athena: ListExecutors
	Athena: ListNamedQueries
	Athena: ListNotebookMetadata
	Athena: ListNotebookSessions
	Athena: ListPreparedStatements
	Athena: ListQueryExecutions
	Athena: ListSessions
	Athena: ListTableMetadata
	Athena: ListWorkGroups
	Athena: PutCapacityAssignmentConfiguration
	Athena: StartCalculationExecution
	Athena: StartQueryExecution
	Athena: StartSession
	Athena: StopCalculationExecution
	Athena: StopQueryExecution
	Athena: TerminateSession
	Athena: UpdateCapacityReservation
	Athena: UpdateDataCatalog

Awalan layanan	Tindakan
	Athena: UpdateNamedQuery Athena: UpdateNotebook Athena: UpdateNotebookMetadata Athena: UpdatePreparedStatement Athena: UpdateWorkGroup

Awalan layanan	Tindakan
Pengelola Audit	Manajer Audit: AssociateAssessmentReportEvidenceFolder
	Manajer Audit: BatchAssociateAssessmentReportEvidence
	Manajer Audit: BatchCreateDelegationByAssessment
	Manajer Audit: BatchDeleteDelegationByAssessment
	Manajer Audit: BatchDisassociateAssessmentReportEvidence
	Manajer Audit: BatchImportEvidenceToAssessmentControl
	Manajer Audit: CreateAssessment
	Manajer Audit: CreateAssessmentFramework
	Manajer Audit: CreateAssessmentReport
	Manajer Audit: CreateControl
	Manajer Audit: DeleteAssessment
	Manajer Audit: DeleteAssessmentFramework
	Manajer Audit: DeleteAssessmentFrameworkShare
	Manajer Audit: DeleteAssessmentReport
	Manajer Audit: DeleteControl
	Manajer Audit: DeregisterAccount
	Manajer Audit: DeregisterOrganizationAdminAccount
	Manajer Audit: DisassociateAssessmentReportEvidenceFolder
	Manajer Audit: GetAccountStatus
	Manajer Audit: GetAssessment
	Manajer Audit: GetAssessmentFramework

Awalan layanan	Tindakan
	Manajer Audit: GetAssessmentReportUrl Manajer Audit: GetChangeLogs Manajer Audit: GetControl Manajer Audit: GetDelegations Manajer Audit: GetEvidence Manajer Audit: GetEvidenceByEvidenceFolder Manajer Audit: GetEvidenceFileUploadUrl Manajer Audit: GetEvidenceFolder Manajer Audit: GetEvidenceFoldersByAssessment Manajer Audit: GetEvidenceFoldersByAssessmentControl Manajer Audit: GetInsights Manajer Audit: GetInsightsByAssessment Manajer Audit: GetOrganizationAdminAccount Manajer Audit: GetServicesInScope Manajer Audit: GetSettings Manajer Audit: ListAssessmentControlInsightsByControlDomain Manajer Audit: ListAssessmentFrameworkShareRequests Manajer Audit: ListAssessmentFrameworks Manajer Audit: ListAssessmentReports Manajer Audit: ListAssessments Manajer Audit: ListControlDomainInsights

Awalan layanan	Tindakan
	Manajer Audit: ListControlDomainInsightsByAssessment Manajer Audit: ListControlInsightsByControlDomain Manajer Audit: ListControls Manajer Audit: ListKeywordsForDataSource Manajer Audit: ListNotifications Manajer Audit: RegisterAccount Manajer Audit: RegisterOrganizationAdminAccount Manajer Audit: StartAssessmentFrameworkShare Manajer Audit: UpdateAssessment Manajer Audit: UpdateAssessmentControl Manajer Audit: UpdateAssessmentControlSetStatus Manajer Audit: UpdateAssessmentFramework Manajer Audit: UpdateAssessmentFrameworkShare Manajer Audit: UpdateAssessmentStatus Manajer Audit: UpdateControl Manajer Audit: UpdateSettings Manajer Audit: ValidateAssessmentReportIntegrity

Awalan layanan	Tindakan
penskalaan otomatis	penskalaan otomatis: AttachInstances
	penskalaan otomatis: AttachLoadBalancerTargetGroups
	penskalaan otomatis: AttachLoadBalancers
	penskalaan otomatis: AttachTrafficSources
	penskalaan otomatis: BatchDeleteScheduledAction
	penskalaan otomatis: BatchPutScheduledUpdateGroupAction
	penskalaan otomatis: CancelInstanceRefresh
	penskalaan otomatis: CompleteLifecycleAction
	penskalaan otomatis: CreateAutoScalingGroup
	penskalaan otomatis: CreateLaunchConfiguration
	penskalaan otomatis: DeleteAutoScalingGroup
	penskalaan otomatis: DeleteLaunchConfiguration
	penskalaan otomatis: DeleteLifecycleHook
	penskalaan otomatis: DeleteNotificationConfiguration
	penskalaan otomatis: DeletePolicy
	penskalaan otomatis: DeleteScheduledAction
	penskalaan otomatis: DeleteWarmPool
	penskalaan otomatis: DescribeAccountLimits
	penskalaan otomatis: DescribeAdjustmentTypes
	penskalaan otomatis: DescribeAutoScalingGroups
	penskalaan otomatis: DescribeAutoScalingInstances

Awalan layanan	Tindakan
	penskalaan otomatis: DescribeAutoScalingNotificationTypes
	penskalaan otomatis: DescribeInstanceRefreshes
	penskalaan otomatis: DescribeLaunchConfigurations
	penskalaan otomatis: DescribeLifecycleHookTypes
	penskalaan otomatis: DescribeLifecycleHooks
	penskalaan otomatis: DescribeLoadBalancerTargetGroups
	penskalaan otomatis: DescribeLoadBalancers
	penskalaan otomatis: DescribeMetricCollectionTypes
	penskalaan otomatis: DescribeNotificationConfigurations
	penskalaan otomatis: DescribePolicies
	penskalaan otomatis: DescribeScalingActivities
	penskalaan otomatis: DescribeScalingProcessTypes
	penskalaan otomatis: DescribeScheduledActions
	penskalaan otomatis: DescribeTerminationPolicyTypes
	penskalaan otomatis: DescribeTrafficSources
	penskalaan otomatis: DescribeWarmPool
	penskalaan otomatis: DetachInstances
	penskalaan otomatis: DetachLoadBalancerTargetGroups
	penskalaan otomatis: DetachLoadBalancers
	penskalaan otomatis: DetachTrafficSources
	penskalaan otomatis: DisableMetricsCollection

Awalan layanan	Tindakan
	penskalaan otomatis: EnableMetricsCollection
	penskalaan otomatis: EnterStandby
	penskalaan otomatis: ExecutePolicy
	penskalaan otomatis: ExitStandby
	penskalaan otomatis: GetPredictiveScalingForecast
	penskalaan otomatis: PutLifecycleHook
	penskalaan otomatis: PutNotificationConfiguration
	penskalaan otomatis: PutScalingPolicy
	penskalaan otomatis: PutScheduledUpdateGroupAction
	penskalaan otomatis: PutWarmPool
	penskalaan otomatis: RecordLifecycleActionHeartbeat
	penskalaan otomatis: ResumeProcesses
	penskalaan otomatis: RollbackInstanceRefresh
	penskalaan otomatis: SetDesiredCapacity
	penskalaan otomatis: SetInstanceHealth
	penskalaan otomatis: SetInstanceProtection
	penskalaan otomatis: StartInstanceRefresh
	penskalaan otomatis: SuspendProcesses
	penskalaan otomatis: TerminateInstanceInAutoScalingGroup
	penskalaan otomatis: UpdateAutoScalingGroup
aws-marketplace	aws-pasar: GetEntitlements

Awalan layanan	Tindakan
pencadangan	cadangan: CancelLegalHold
	cadangan: CreateBackupPlan
	cadangan: CreateBackupSelection
	cadangan: CreateBackupVault
	cadangan: CreateFramework
	cadangan: CreateLegalHold
	cadangan: CreateLogicallyAirGappedBackupVault
	cadangan: CreateReportPlan
	cadangan: CreateRestoreAccessBackupVault
	cadangan: CreateRestoreTestingPlan
	cadangan: CreateRestoreTestingSelection
	cadangan: CreateTieringConfiguration
	cadangan: DeleteBackupPlan
	cadangan: DeleteBackupSelection
	cadangan: DeleteBackupVault
	cadangan: DeleteBackupVaultAccessPolicy
	cadangan: DeleteBackupVaultLockConfiguration
	cadangan: DeleteBackupVaultNotifications
	cadangan: DeleteFramework
	cadangan: DeleteRecoveryPoint
	cadangan: DeleteReportPlan

Awalan layanan	Tindakan
	cadangan: DeleteRestoreTestingPlan
	cadangan: DeleteRestoreTestingSelection
	cadangan: DeleteTieringConfiguration
	cadangan: DescribeBackupJob
	cadangan: DescribeBackupVault
	cadangan: DescribeCopyJob
	cadangan: DescribeFramework
	cadangan: DescribeGlobalSettings
	cadangan: DescribeProtectedResource
	cadangan: DescribeRecoveryPoint
	cadangan: DescribeRegionSettings
	cadangan: DescribeReportJob
	cadangan: DescribeReportPlan
	cadangan: DescribeRestoreJob
	cadangan: DescribeScanJob
	cadangan: DisassociateRecoveryPoint
	cadangan: DisassociateRecoveryPointFromParent
	cadangan: ExportBackupPlanTemplate
	cadangan: GetBackupPlan
	cadangan: GetBackupPlanFromJSON
	cadangan: GetBackupPlanFromTemplate

Awalan layanan	Tindakan
	cadangan: GetBackupSelection
	cadangan: GetBackupVaultAccessPolicy
	cadangan: GetBackupVaultNotifications
	cadangan: GetLegalHold
	cadangan: GetRecoveryPointRestoreMetadata
	cadangan: GetRestoreJobMetadata
	cadangan: GetRestoreTestingInferredMetadata
	cadangan: GetRestoreTestingPlan
	cadangan: GetRestoreTestingSelection
	cadangan: GetSupportedResourceTypes
	cadangan: GetTieringConfiguration
	cadangan: ListBackupJobSummaries
	cadangan: ListBackupJobs
	cadangan: ListBackupPlanTemplates
	cadangan: ListBackupPlanVersions
	cadangan: ListBackupPlans
	cadangan: ListBackupSelections
	cadangan: ListBackupVaults
	cadangan: ListCopyJobSummaries
	cadangan: ListCopyJobs
	cadangan: ListFrameworks

Awalan layanan	Tindakan
	cadangan: ListIndexedRecoveryPoints
	cadangan: ListLegalHolds
	cadangan: ListProtectedResources
	cadangan: ListRecoveryPointsByBackupVault
	cadangan: ListRecoveryPointsByLegalHold
	cadangan: ListRecoveryPointsByResource
	cadangan: ListReportJobs
	cadangan: ListReportPlans
	cadangan: ListRestoreAccessBackupVaults
	cadangan: ListRestoreJobSummaries
	cadangan: ListRestoreJobs
	cadangan: ListRestoreJobsByProtectedResource
	cadangan: ListRestoreTestingPlans
	cadangan: ListRestoreTestingSelections
	cadangan: ListScanJobSummaries
	cadangan: ListScanJobs
	cadangan: ListTieringConfigurations
	cadangan: PutBackupVaultAccessPolicy
	cadangan: PutBackupVaultLockConfiguration
	cadangan: PutBackupVaultNotifications
	cadangan: PutRestoreValidationResult

Awalan layanan	Tindakan
	cadangan: StartBackupJob
	cadangan: StartCopyJob
	cadangan: StartReportJob
	cadangan: StartRestoreJob
	cadangan: StopBackupJob
	cadangan: UpdateBackupPlan
	cadangan: UpdateFramework
	cadangan: UpdateGlobalSettings
	cadangan: UpdateRecoveryPointLifecycle
	cadangan: UpdateRegionSettings
	cadangan: UpdateReportPlan
	cadangan: UpdateRestoreTestingPlan
	cadangan: UpdateRestoreTestingSelection
	cadangan: UpdateTieringConfiguration

Awalan layanan	Tindakan
batch	batch: CancelJob batch: CreateComputeEnvironment batch: CreateConsumableResource batch: CreateJobQueue batch: CreateQuotaShare batch: CreateSchedulingPolicy batch: CreateServiceEnvironment batch: DeleteComputeEnvironment batch: DeleteConsumableResource batch: DeleteJobQueue batch: DeleteQuotaShare batch: DeleteSchedulingPolicy batch: DeleteServiceEnvironment batch: DeregisterJobDefinition batch: DescribeComputeEnvironments batch: DescribeConsumableResource batch: DescribeJobDefinitions batch: DescribeJobQueues batch: DescribeJobs batch: DescribeQuotaShare batch: DescribeSchedulingPolicies

Awalan layanan	Tindakan
	batch: DescribeServiceEnvironments batch: DescribeServiceJob batch: GetJobQueueSnapshot batch: ListConsumableResources batch: ListJobs batch: ListJobsByConsumableResource batch: ListQuotaShares batch: ListSchedulingPolicies batch: ListServiceJobs batch: RegisterJobDefinition batch: SubmitJob batch: SubmitServiceJob batch: TerminateJob batch: TerminateServiceJob batch: UpdateComputeEnvironment batch: UpdateConsumableResource batch: UpdateJobQueue batch: UpdateQuotaShare batch: UpdateSchedulingPolicy batch: UpdateServiceEnvironment batch: UpdateServiceJob

Awalan layanan	Tindakan
braket	Braket: CancelJob
	Braket: CancelQuantumTask
	Braket: CreateJob
	Braket: CreateQuantumTask
	Braket: CreateSpendingLimit
	Braket: GetDevice
	Braket: GetJob
	Braket: GetQuantumTask
	Braket: SearchDevices
	Braket: SearchJobs
	Braket: SearchQuantumTasks
	Braket: SearchSpendingLimits

Awalan layanan	Tindakan
anggaran	anggaran: CreateBudgetAction anggaran: DeleteBudgetAction anggaran: DescribeBudgetAction anggaran: DescribeBudgetActionHistories anggaran: DescribeBudgetActionsForAccount anggaran: DescribeBudgetActionsForBudget anggaran: ExecuteBudgetAction anggaran: ModifyBudget anggaran: UpdateBudgetAction anggaran: ViewBudget
awan9	cloud9: EC2 CreateEnvironment awan9: CreateEnvironmentMembership awan9: DeleteEnvironment awan9: DeleteEnvironmentMembership awan9: DescribeEnvironmentMemberships awan9: DescribeEnvironmentStatus awan9: DescribeEnvironments awan9: ListEnvironments awan9: UpdateEnvironment awan9: UpdateEnvironmentMembership

Awalan layanan	Tindakan
pembentukan awan	pembentukan awan: BatchDescribeTypeConfigurations pembentukan awan: CancelUpdateStack pembentukan awan: ContinueUpdateRollback pembentukan awan: CreateChangeSet pembentukan awan: CreateGeneratedTemplate pembentukan awan: CreateStack pembentukan awan: CreateStackInstances pembentukan awan: CreateStackSet pembentukan awan: DeactivateType pembentukan awan: DeleteChangeSet pembentukan awan: DeleteGeneratedTemplate pembentukan awan: DeleteStack pembentukan awan: DeleteStackInstances pembentukan awan: DeleteStackSet pembentukan awan: DeregisterType pembentukan awan: DescribeAccountLimits pembentukan awan: DescribeChangeSet pembentukan awan: DescribeChangeSetHooks pembentukan awan: DescribeEvents pembentukan awan: DescribeGeneratedTemplate pembentukan awan: DescribeOrganizationsAccess

Awalan layanan	Tindakan
	pembentukan awan: DescribePublisher
	pembentukan awan: DescribeResourceScan
	pembentukan awan: DescribeStackDriftDetectionStatus
	pembentukan awan: DescribeStackEvents
	pembentukan awan: DescribeStackInstance
	pembentukan awan: DescribeStackResource
	pembentukan awan: DescribeStackResourceDrifts
	pembentukan awan: DescribeStackResources
	pembentukan awan: DescribeStackSet
	pembentukan awan: DescribeStackSetOperation
	pembentukan awan: DescribeStacks
	pembentukan awan: DescribeType
	pembentukan awan: DescribeTypeRegistration
	pembentukan awan: DetectStackDrift
	pembentukan awan: DetectStackResourceDrift
	pembentukan awan: DetectStackSetDrift
	pembentukan awan: EstimateTemplateCost
	pembentukan awan: ExecuteChangeSet
	pembentukan awan: GetGeneratedTemplate
	pembentukan awan: GetHookResult
	pembentukan awan: GetStackPolicy

Awalan layanan	Tindakan
	pembentukan awan: GetTemplate pembentukan awan: GetTemplateSummary pembentukan awan: ImportStacksToStackSet pembentukan awan: ListChangeSets pembentukan awan: ListExports pembentukan awan: ListGeneratedTemplates pembentukan awan: ListHookResults pembentukan awan: ListImports pembentukan awan: ListResourceScanRelatedResources pembentukan awan: ListResourceScanResources pembentukan awan: ListResourceScans pembentukan awan: ListStackInstanceResourceDrifts pembentukan awan: ListStackInstances pembentukan awan: ListStackRefactors pembentukan awan: ListStackResources pembentukan awan: ListStackSetAutoDeploymentTargets pembentukan awan: ListStackSetOperationResults pembentukan awan: ListStackSetOperations pembentukan awan: ListStackSets pembentukan awan: ListTypeRegistrations pembentukan awan: ListTypeVersions

Awalan layanan	Tindakan
	pembentukan awan: ListTypes pembentukan awan: PublishType pembentukan awan: RecordHandlerProgress pembentukan awan: RegisterPublisher pembentukan awan: RegisterType pembentukan awan: RollbackStack pembentukan awan: SetStackPolicy pembentukan awan: SetTypeConfiguration pembentukan awan: SetTypeDefaultVersion pembentukan awan: SignalResource pembentukan awan: StartResourceScan pembentukan awan: StopStackSetOperation pembentukan awan: TestType pembentukan awan: UpdateGeneratedTemplate pembentukan awan: UpdateStack pembentukan awan: UpdateStackInstances pembentukan awan: UpdateStackSet pembentukan awan: UpdateTerminationProtection pembentukan awan: ValidateTemplate

Awalan layanan	Tindakan
cloudfront	cloudfront: AssociateAlias
	cloudfront: AssociateDistributionTenantWebACL
	cloudfront: AssociateDistributionWebACL
	cloudfront: CreateCachePolicy
	cloudfront: CreateCloudFrontOriginAccessIdentity
	cloudfront: CreateConnectionFunction
	cloudfront: CreateContinuousDeploymentPolicy
	cloudfront: CreateDistributionTenant
	cloudfront: CreateFieldLevelEncryptionConfig
	cloudfront: CreateFieldLevelEncryptionProfile
	cloudfront: CreateFunction
	cloudfront: CreateInvalidation
	cloudfront: CreateKeyGroup
	cloudfront: CreateKeyValueStore
	cloudfront: CreateMonitoringSubscription
	cloudfront: CreateOriginAccessControl
	cloudfront: CreateOriginRequestPolicy
	cloudfront: CreatePublicKey
	cloudfront: CreateRealtimeLogConfig
	cloudfront: CreateResponseHeadersPolicy
	cloudfront: CreateTrustStore

Awalan layanan	Tindakan
	cloudfront: DeleteAnycastIpList
	cloudfront: DeleteCachePolicy
	cloudfront: DeleteCloudFrontOriginAccessIdentity
	cloudfront: DeleteConnectionFunction
	cloudfront: DeleteConnectionGroup
	cloudfront: DeleteContinuousDeploymentPolicy
	cloudfront: DeleteDistribution
	cloudfront: DeleteDistributionTenant
	cloudfront: DeleteFieldLevelEncryptionConfig
	cloudfront: DeleteFieldLevelEncryptionProfile
	cloudfront: DeleteFunction
	cloudfront: DeleteKeyGroup
	cloudfront: DeleteKeyValueStore
	cloudfront: DeleteMonitoringSubscription
	cloudfront: DeleteOriginAccessControl
	cloudfront: DeleteOriginRequestPolicy
	cloudfront: DeletePublicKey
	cloudfront: DeleteRealtimeLogConfig
	cloudfront: DeleteResponseHeadersPolicy
	cloudfront: DeleteStreamingDistribution
	cloudfront: DeleteTrustStore

Awalan layanan	Tindakan
	cloudfront: DeleteVpcOrigin
	cloudfront: DescribeFunction
	cloudfront: DescribeKeyValueStore
	cloudfront: DisassociateDistributionTenantWebACL
	cloudfront: DisassociateDistributionWebACL
	cloudfront: GetAnycastIpList
	cloudfront: GetCachePolicy
	cloudfront: GetCachePolicyConfig
	cloudfront: GetCloudFrontOriginAccessIdentity
	cloudfront: GetCloudFrontOriginAccessIdentityConfig
	cloudfront: GetContinuousDeploymentPolicy
	cloudfront: GetContinuousDeploymentPolicyConfig
	cloudfront: GetDistributionConfig
	cloudfront: GetFieldLevelEncryption
	cloudfront: GetFieldLevelEncryptionConfig
	cloudfront: GetFieldLevelEncryptionProfile
	cloudfront: GetFieldLevelEncryptionProfileConfig
	cloudfront: GetFunction
	cloudfront: GetInvalidation
	cloudfront: GetInvalidationForDistributionTenant
	cloudfront: GetKeyGroup

Awalan layanan	Tindakan
	cloudfront: GetKeyGroupConfig
	cloudfront: GetMonitoringSubscription
	cloudfront: GetOriginAccessControl
	cloudfront: GetOriginAccessControlConfig
	cloudfront: GetOriginRequestPolicy
	cloudfront: GetOriginRequestPolicyConfig
	cloudfront: GetPublicKey
	cloudfront: GetPublicKeyConfig
	cloudfront: GetRealtimeLogConfig
	cloudfront: GetResponseHeadersPolicy
	cloudfront: GetResponseHeadersPolicyConfig
	cloudfront: GetStreamingDistribution
	cloudfront: GetStreamingDistributionConfig
	cloudfront: GetVpcOrigin
	cloudfront: ListAnycastIpLists
	cloudfront: ListCachePolicies
	cloudfront: ListCloudFrontOriginAccessIdentities
	cloudfront: ListConflictingAliases
	cloudfront: ListConnectionFunctions
	cloudfront: ListConnectionGroups
	cloudfront: ListContinuousDeploymentPolicies

Awalan layanan	Tindakan
	cloudfront: ListDistributionTenants
	cloudfront: ListDistributionTenantsByCustomization
	cloudfront: ListDistributions
	cloudfront: ListDistributionsByAnycastIpListId
	cloudfront: ListDistributionsByCachePolicyId
	cloudfront: ListDistributionsByConnectionMode
	cloudfront: ListDistributionsByKeyGroup
	cloudfront: ListDistributionsByOriginRequestPolicyId
	cloudfront: ListDistributionsByRealtimeLogConfig
	cloudfront: ListDistributionsByResponseHeadersPolicyId
	cloudfront: ListDistributionsByVpcOriginId
	cloudfront: ListDistributionsByWebACLId
	cloudfront: ListFieldLevelEncryptionConfigs
	cloudfront: ListFieldLevelEncryptionProfiles
	cloudfront: ListFunctions
	cloudfront: ListInvalidations
	cloudfront: ListInvalidationsForDistributionTenant
	cloudfront: ListKeyGroups
	cloudfront: ListKeyValueStores
	cloudfront: ListOriginAccessControls
	cloudfront: ListOriginRequestPolicies

Awalan layanan	Tindakan
	cloudfront: ListPublicKeys
	cloudfront: ListRealtimeLogConfigs
	cloudfront: ListResponseHeadersPolicies
	cloudfront: ListStreamingDistributions
	cloudfront: ListTrustStores
	cloudfront: PublishConnectionFunction
	cloudfront: PublishFunction
	cloudfront: TestConnectionFunction
	cloudfront: TestFunction
	cloudfront: UpdateAnycastIpList
	cloudfront: UpdateCachePolicy
	cloudfront: UpdateCloudFrontOriginAccessIdentity
	cloudfront: UpdateConnectionFunction
	cloudfront: UpdateConnectionGroup
	cloudfront: UpdateContinuousDeploymentPolicy
	cloudfront: UpdateDistribution
	cloudfront: UpdateDistributionTenant
	cloudfront: UpdateFieldLevelEncryptionConfig
	cloudfront: UpdateFieldLevelEncryptionProfile
	cloudfront: UpdateFunction
	cloudfront: UpdateKeyGroup

Awalan layanan	Tindakan
	cloudfront: UpdateKeyValueStore cloudfront: UpdateOriginAccessControl cloudfront: UpdateOriginRequestPolicy cloudfront: UpdatePublicKey cloudfront: UpdateRealtimeLogConfig cloudfront: UpdateResponseHeadersPolicy cloudfront: UpdateTrustStore
cloudhsm	cloudhsm: CreateHsm cloudhsm: DeleteBackup cloudhsm: DeleteHsm cloudhsm: DeleteResourcePolicy cloudhsm: DescribeBackups cloudhsm: DescribeClusters cloudhsm: GetResourcePolicy cloudhsm: InitializeCluster cloudhsm: ModifyBackupAttributes cloudhsm: ModifyCluster cloudhsm: PutResourcePolicy cloudhsm: RestoreBackup

Awalan layanan	Tindakan
cloudsearch	cloudsearch: BuildSuggesters
	cloudsearch: CreateDomain
	cloudsearch: DefineAnalysisScheme
	cloudsearch: DefineExpression
	cloudsearch: DefineIndexField
	cloudsearch: DefineSuggester
	cloudsearch: DeleteAnalysisScheme
	cloudsearch: DeleteDomain
	cloudsearch: DeleteExpression
	cloudsearch: DeleteIndexField
	cloudsearch: DeleteSuggester
	cloudsearch: DescribeAnalysisSchemes
	cloudsearch: DescribeAvailabilityOptions
	cloudsearch: DescribeDomainEndpointOptions
	cloudsearch: DescribeDomains
	cloudsearch: DescribeExpressions
	cloudsearch: DescribeIndexFields
	cloudsearch: DescribeScalingParameters
	cloudsearch: DescribeServiceAccessPolicies
	cloudsearch: DescribeSuggesters
	cloudsearch: IndexDocuments

Awalan layanan	Tindakan
	cloudsearch: ListDomainNames cloudsearch: UpdateAvailabilityOptions cloudsearch: UpdateDomainEndpointOptions cloudsearch: UpdateScalingParameters cloudsearch: UpdateServiceAccessPolicies

Awalan layanan	Tindakan
cloudtrail	cloudtrail: CancelQuery cloudtrail: CreateChannel cloudtrail: CreateDashboard cloudtrail: CreateEventDataStore cloudtrail: CreateTrail cloudtrail: DeleteChannel cloudtrail: DeleteDashboard cloudtrail: DeleteEventDataStore cloudtrail: DeleteResourcePolicy cloudtrail: DeleteTrail cloudtrail: DeregisterOrganizationDelegatedAdmin cloudtrail: DescribeQuery cloudtrail: DescribeTrails cloudtrail: DisableFederation cloudtrail: GenerateQuery cloudtrail: GetChannel cloudtrail: GetDashboard cloudtrail: GetEventConfiguration cloudtrail: GetEventDataStore cloudtrail: GetEventDataStoreData cloudtrail: GetEventSelectors

Awalan layanan	Tindakan
	cloudtrail: GetImport cloudtrail: GetInsightSelectors cloudtrail: GetResourcePolicy cloudtrail: GetTrail cloudtrail: GetTrailStatus cloudtrail: ListChannels cloudtrail: ListDashboards cloudtrail: ListEventDataStores cloudtrail: ListImportFailures cloudtrail: ListImports cloudtrail: ListInsightsData cloudtrail: ListPublicKeys cloudtrail: ListQueries cloudtrail: ListTrails cloudtrail: LookupEvents cloudtrail: PutEventConfiguration cloudtrail: PutEventSelectors cloudtrail: PutInsightSelectors cloudtrail: PutResourcePolicy cloudtrail: RegisterOrganizationDelegatedAdmin cloudtrail: RestoreEventDataStore

Awalan layanan	Tindakan
	cloudtrail: SearchSampleQueries cloudtrail: StartEventDataStoreIngestion cloudtrail: StartImport cloudtrail: StartLogging cloudtrail: StartQuery cloudtrail: StopEventDataStoreIngestion cloudtrail: StopImport cloudtrail: StopLogging cloudtrail: UpdateChannel cloudtrail: UpdateDashboard cloudtrail: UpdateEventDataStore cloudtrail: UpdateTrail

Awalan layanan	Tindakan
cloudwatch	jam tangan awan: DeleteAlarmMuteRule
	jam tangan awan: DeleteAlarms
	jam tangan awan: DeleteAnomalyDetector
	jam tangan awan: DeleteDashboards
	jam tangan awan: DeleteInsightRules
	jam tangan awan: DeleteMetricStream
	jam tangan awan: DescribeAlarmHistory
	jam tangan awan: DescribeAlarms
	jam tangan awan: DescribeAlarmsForMetric
	jam tangan awan: DescribeAnomalyDetectors
	jam tangan awan: DescribeInsightRules
	jam tangan awan: DisableAlarmActions
	jam tangan awan: DisableInsightRules
	jam tangan awan: EnableAlarmActions
	jam tangan awan: EnableInsightRules
	jam tangan awan: GetAlarmMuteRule
	jam tangan awan: GetDashboard
	jam tangan awan: GetDataset
	jam tangan awan: GetInsightRuleReport
	jam tangan awan: GetMetricStatistics
	jam tangan awan: GetMetricStream

Awalan layanan	Tindakan
	jam tangan awan: ListAlarmMuteRules
	jam tangan awan: ListDashboards
	jam tangan awan: ListManagedInsightRules
	jam tangan awan: ListMetricStreams
	jam tangan awan: PutAlarmMuteRule
	jam tangan awan: PutAnomalyDetector
	jam tangan awan: PutCompositeAlarm
	jam tangan awan: PutDashboard
	jam tangan awan: PutInsightRule
	jam tangan awan: PutLogAlarm
	jam tangan awan: PutManagedInsightRules
	jam tangan awan: PutMetricAlarm
	jam tangan awan: PutMetricStream
	jam tangan awan: SetAlarmState
	jam tangan awan: StartMetricStreams
	jam tangan awan: StopMetricStreams

Awalan layanan	Tindakan
kodeartefak	codeartefak: AssociateExternalConnection codeartefak: CopyPackageVersions codeartefak: CreateDomain codeartefak: CreateRepository codeartefak: DeleteDomain codeartefak: DeleteDomainPermissionsPolicy codeartefak: DeletePackage codeartefak: DeletePackageVersions codeartefak: DeleteRepository codeartefak: DeleteRepositoryPermissionsPolicy codeartefak: DescribeDomain codeartefak: DescribePackage codeartefak: DescribePackageVersion codeartefak: DescribeRepository codeartefak: DisassociateExternalConnection codeartefak: DisposePackageVersions codeartefak: GetAssociatedPackageGroup codeartefak: GetAuthorizationToken codeartefak: GetDomainPermissionsPolicy codeartefak: GetPackageVersionAsset codeartefak: GetPackageVersionReadme

Awalan layanan	Tindakan
	codeartefak: GetRepositoryEndpoint codeartefak: GetRepositoryPermissionsPolicy codeartefak: ListDomains codeartefak: ListPackageGroups codeartefak: ListPackageVersionAssets codeartefak: ListPackageVersionDependencies codeartefak: ListPackageVersions codeartefak: ListPackages codeartefak: ListRepositories codeartefak: ListRepositoriesInDomain codeartefak: PublishPackageVersion codeartefak: PutDomainPermissionsPolicy codeartefak: PutPackageMetadata codeartefak: PutPackageOriginConfiguration codeartefak: PutRepositoryPermissionsPolicy codeartefak: ReadFromRepository codeartefak: UpdatePackageVersionsStatus codeartefak: UpdateRepository

Awalan layanan	Tindakan
penyebaran kode	penyebaran kode: BatchGetApplicationRevisions
	penyebaran kode: BatchGetApplications
	penyebaran kode: BatchGetDeploymentGroups
	penyebaran kode: BatchGetDeploymentInstances
	penyebaran kode: BatchGetDeploymentTargets
	penyebaran kode: BatchGetDeployments
	penyebaran kode: BatchGetOnPremisesInstances
	penyebaran kode: ContinueDeployment
	penyebaran kode: CreateApplication
	penyebaran kode: CreateDeployment
	penyebaran kode: CreateDeploymentConfig
	penyebaran kode: CreateDeploymentGroup
	penyebaran kode: DeleteApplication
	penyebaran kode: DeleteDeploymentConfig
	penyebaran kode: DeleteDeploymentGroup
	penyebaran kode: DeleteGitHubAccountToken
	penyebaran kode: DeleteResourcesByExternalId
	penyebaran kode: DeregisterOnPremisesInstance
	penyebaran kode: GetApplication
	penyebaran kode: GetApplicationRevision
	penyebaran kode: GetDeployment

Awalan layanan	Tindakan
	penyebaran kode: GetDeploymentConfig
	penyebaran kode: GetDeploymentGroup
	penyebaran kode: GetDeploymentInstance
	penyebaran kode: GetDeploymentTarget
	penyebaran kode: GetOnPremisesInstance
	penyebaran kode: ListApplicationRevisions
	penyebaran kode: ListApplications
	penyebaran kode: ListDeploymentConfigs
	penyebaran kode: ListDeploymentGroups
	penyebaran kode: ListDeploymentInstances
	penyebaran kode: ListDeploymentTargets
	penyebaran kode: ListDeployments
	penyebaran kode: ListGitHubAccountTokenNames
	penyebaran kode: ListOnPremisesInstances
	penyebaran kode: PutLifecycleEventHookExecutionStatus
	penyebaran kode: RegisterApplicationRevision
	penyebaran kode: RegisterOnPremisesInstance
	penyebaran kode: SkipWaitTimeForInstanceTermination
	penyebaran kode: StopDeployment
	penyebaran kode: UpdateApplication
	penyebaran kode: UpdateDeploymentGroup

Awalan layanan	Tindakan
codeguru-profiler	codeguru-profiler: AddNotificationChannels codeguru-profiler: BatchGetFrameMetricData codeguru-profiler: CreateProfilingGroup codeguru-profiler: DeleteProfilingGroup codeguru-profiler: DescribeProfilingGroup codeguru-profiler: GetFindingsReportAccountSummary codeguru-profiler: GetNotificationConfiguration codeguru-profiler: GetPolicy codeguru-profiler: GetProfile codeguru-profiler: GetRecommendations codeguru-profiler: ListFindingsReports codeguru-profiler: ListProfileTimes codeguru-profiler: ListProfilingGroups codeguru-profiler: PutPermission codeguru-profiler: RemoveNotificationChannel codeguru-profiler: RemovePermission codeguru-profiler: SubmitFeedback codeguru-profiler: UpdateProfilingGroup

Awalan layanan	Tindakan
pengulas codeguru	pengulas codeguru: AssociateRepository
	pengulas codeguru: CreateCodeReview
	pengulas codeguru: DescribeCodeReview
	pengulas codeguru: DescribeRecommendationFeedback
	pengulas codeguru: DescribeRepositoryAssociation
	pengulas codeguru: DisassociateRepository
	pengulas codeguru: ListCodeReviews
	pengulas codeguru: ListRecommendationFeedback
	pengulas codeguru: ListRecommendations
	pengulas codeguru: ListRepositoryAssociations
	pengulas codeguru: PutRecommendationFeedback

Awalan layanan	Tindakan
codepipeline	codepipeline: AcknowledgeJob
	codepipeline: AcknowledgeThirdPartyJob
	codepipeline: CreateCustomActionType
	codepipeline: CreatePipeline
	codepipeline: DeleteCustomActionType
	codepipeline: DeletePipeline
	codepipeline: DeleteWebhook
	codepipeline: DeregisterWebhookWithThirdParty
	codepipeline: GetActionType
	codepipeline: GetJobDetails
	codepipeline: GetPipeline
	codepipeline: GetPipelineExecution
	codepipeline: GetPipelineState
	codepipeline: GetThirdPartyJobDetails
	codepipeline: ListActionExecutions
	codepipeline: ListActionTypes
	codepipeline: ListPipelineExecutions
	codepipeline: ListPipelines
	codepipeline: ListRuleExecutions
	codepipeline: ListRuleTypes
	codepipeline: ListWebhooks

Awalan layanan	Tindakan
	codepipeline: OverrideStageCondition codepipeline: PollForJobs codepipeline: PollForThirdPartyJobs codepipeline: PutActionRevision codepipeline: PutApprovalResult codepipeline: PutJobFailureResult codepipeline: PutJobSuccessResult codepipeline: PutThirdPartyJobFailureResult codepipeline: PutThirdPartyJobSuccessResult codepipeline: PutWebhook codepipeline: RegisterWebhookWithThirdParty codepipeline: RollbackStage codepipeline: StartPipelineExecution codepipeline: StopPipelineExecution codepipeline: UpdateActionType codepipeline: UpdatePipeline

Awalan layanan	Tindakan
bintang kode	bintang kode: AssociateTeamMember bintang kode: CreateProject bintang kode: CreateUserProfile bintang kode: DeleteProject bintang kode: DeleteUserProfile bintang kode: DescribeProject bintang kode: DescribeUserProfile bintang kode: DisassociateTeamMember bintang kode: ListProjects bintang kode: ListResources bintang kode: ListTeamMembers bintang kode: ListUserProfiles bintang kode: UpdateProject bintang kode: UpdateTeamMember bintang kode: UpdateUserProfile

Awalan layanan	Tindakan
pemberitahuan bintang kode	pemberitahuan bintang kode: CreateNotificationRule pemberitahuan bintang kode: DeleteNotificationRule pemberitahuan bintang kode: DeleteTarget pemberitahuan bintang kode: DescribeNotificationRule pemberitahuan bintang kode: ListEventTypes pemberitahuan bintang kode: ListNotificationRules pemberitahuan bintang kode: ListTargets Codestar-notifikasi:Berlangganan Codestar-notifications:berhenti berlangganan pemberitahuan bintang kode: UpdateNotificationRule

Awalan layanan	Tindakan
kognito-identitas	kognito-identitas: CreateIdentityPool
	kognito-identitas: DeleteIdentities
	kognito-identitas: DeleteIdentityPool
	kognito-identitas: DescribeIdentity
	kognito-identitas: DescribeIdentityPool
	kognito-identitas: GetIdentityPoolRoles
	kognito-identitas: ListIdentities
	kognito-identitas: ListIdentityPools
	kognito-identitas: LookupDeveloperIdentity
	kognito-identitas: MergeDeveloperIdentities
	kognito-identitas: SetIdentityPoolRoles
	kognito-identitas: UnlinkDeveloperIdentity
	kognito-identitas: UpdateIdentityPool

Awalan layanan	Tindakan
cognito-idp	cognito-idp: AddCustomAttributes cognito-idp: AddUserPoolClientSecret cognito-idp: AdminAddUserToGroup cognito-idp: AdminConfirmSignUp cognito-idp: AdminCreateUser cognito-idp: AdminDeleteUser cognito-idp: AdminDeleteUserAttributes cognito-idp: AdminDisableProviderForUser cognito-idp: AdminDisableUser cognito-idp: AdminEnableUser cognito-idp: AdminForgetDevice cognito-idp: AdminGetDevice cognito-idp: AdminGetUser cognito-idp: AdminGetUserAuthFactors cognito-idp: AdminInitiateAuth cognito-idp: AdminLinkProviderForUser cognito-idp: AdminListDevices cognito-idp: AdminListGroupsForUser cognito-idp: AdminListUserAuthEvents cognito-idp: AdminRemoveUserFromGroup cognito-idp: AdminResetUserPassword

Awalan layanan	Tindakan
	cognito-idp: AdminRespondToAuthChallenge cognito-idp: AdminSetUserMFAPreference cognito-idp: AdminSetUserPassword cognito-idp: AdminSetUserSettings cognito-idp: AdminUpdateAuthEventFeedback cognito-idp: AdminUpdateDeviceStatus cognito-idp: AdminUpdateUserAttributes cognito-idp: AdminUserGlobalSignOut cognito-idp: AssociateSoftwareToken cognito-idp: ChangePassword cognito-idp: ConfirmDevice cognito-idp: ConfirmForgotPassword cognito-idp: ConfirmSignUp cognito-idp: CreateGroup cognito-idp: CreateIdentityProvider cognito-idp: CreateManagedLoginBranding cognito-idp: CreateResourceServer cognito-idp: CreateTerms cognito-idp: CreateUserImportJob cognito-idp: CreateUserPool cognito-idp: CreateUserPoolClient

Awalan layanan	Tindakan
	cognito-idp: CreateUserPoolDomain cognito-idp: CreateUserPoolReplica cognito-idp: DeleteGroup cognito-idp: DeleteIdentityProvider cognito-idp: DeleteManagedLoginBranding cognito-idp: DeleteResourceServer cognito-idp: DeleteTerms cognito-idp: DeleteUser cognito-idp: DeleteUserAttributes cognito-idp: DeleteUserPool cognito-idp: DeleteUserPoolClient cognito-idp: DeleteUserPoolClientSecret cognito-idp: DeleteUserPoolDomain cognito-idp: DeleteUserPoolReplica cognito-idp: DescribeIdentityProvider cognito-idp: DescribeManagedLoginBranding cognito-idp: DescribeManagedLoginBrandingByClient cognito-idp: DescribeResourceServer cognito-idp: DescribeRiskConfiguration cognito-idp: DescribeTerms cognito-idp: DescribeUserImportJob

Awalan layanan	Tindakan
	cognito-idp: DescribeUserPool cognito-idp: DescribeUserPoolClient cognito-idp: DescribeUserPoolDomain cognito-idp: ForgetDevice cognito-idp: ForgotPassword Cognito-IDP:GetCSVHeader cognito-idp: GetDevice cognito-idp: GetGroup cognito-idp: GetIdentityProviderByIdentifier cognito-idp: GetLogDeliveryConfiguration cognito-idp: GetProvisionedLimit cognito-idp: GetSigningCertificate Cognito-IDP:getuicustomisasi cognito-idp: GetUser cognito-idp: GetUserAttributeVerificationCode cognito-idp: GetUserPoolMfaConfig cognito-idp: GlobalSignOut cognito-idp: InitiateAuth cognito-idp: ListDevices cognito-idp: ListGroups cognito-idp: ListIdentityProviders

Awalan layanan	Tindakan
	cognito-idp: ListResourceServers cognito-idp: ListTerms cognito-idp: ListUserImportJobs cognito-idp: ListUserPoolClientSecrets cognito-idp: ListUserPoolClients cognito-idp: ListUserPoolReplicas cognito-idp: ListUserPools cognito-idp: ListUsers cognito-idp: ListUsersInGroup cognito-idp: ResendConfirmationCode cognito-idp: RespondToAuthChallenge cognito-idp: RevokeToken cognito-idp: SetLogDeliveryConfiguration cognito-idp: SetRiskConfiguration Cognito-IDP: Setuicustomisasi cognito-idp: SetUserMFAPreference cognito-idp: SetUserPoolMfaConfig cognito-idp: SetUserSettings cognito-idp: SignUp cognito-idp: StartUserImportJob cognito-idp: StopUserImportJob

Awalan layanan	Tindakan
	cognito-idp: UpdateAuthEventFeedback cognito-idp: UpdateDeviceStatus cognito-idp: UpdateGroup cognito-idp: UpdateIdentityProvider cognito-idp: UpdateProvisionedLimit cognito-idp: UpdateResourceServer cognito-idp: UpdateTerms cognito-idp: UpdateUserAttributes cognito-idp: UpdateUserPool cognito-idp: UpdateUserPoolClient cognito-idp: UpdateUserPoolDomain cognito-idp: UpdateUserPoolReplica cognito-idp: VerifySoftwareToken cognito-idp: VerifyUserAttribute

Awalan layanan	Tindakan
sinkronisasi kognito	sinkronisasi kognito: BulkPublish
	sinkronisasi kognito: DeleteDataset
	sinkronisasi kognito: DescribeDataset
	sinkronisasi kognito: DescribeIdentityPoolUsage
	sinkronisasi kognito: DescribeIdentityUsage
	sinkronisasi kognito: GetBulkPublishDetails
	sinkronisasi kognito: GetCognitoEvents
	sinkronisasi kognito: GetIdentityPoolConfiguration
	sinkronisasi kognito: ListDatasets
	sinkronisasi kognito: ListIdentityPoolUsage
	sinkronisasi kognito: ListRecords
	sinkronisasi kognito: RegisterDevice
	sinkronisasi kognito: SetCognitoEvents
	sinkronisasi kognito: SetIdentityPoolConfiguration
	sinkronisasi kognito: SubscribeToDataset
	sinkronisasi kognito: UnsubscribeFromDataset
	sinkronisasi kognito: UpdateRecords

Awalan layanan	Tindakan
memahami-medis	memahami medis: DescribeEntitiesDetectionV2Job memahami medis: DescribeICD10CMInferenceJob memahami medis: DescribePHIDetectionJob memahami medis: DescribeRxNormInferenceJob memahami medis: DescribeSNOMEDCTInferenceJob memahami medis: V2 DetectEntities ComprehendMedical: DetectPHI ComprehendMedical: infericd10cm memahami medis: InferRxNorm ComprehendMedical: InfersnomeDCT memahami medis: ListEntitiesDetectionV2Jobs memahami medis: ListICD10CMInferenceJobs memahami medis: ListPHIDetectionJobs memahami medis: ListRxNormInferenceJobs memahami medis: ListSNOMEDCTInferenceJobs memahami medis: StartEntitiesDetectionV2Job memahami medis: StartICD10CMInferenceJob memahami medis: StartPHIDetectionJob memahami medis: StartRxNormInferenceJob memahami medis: StartSNOMEDCTInferenceJob memahami medis: StopEntitiesDetectionV2Job

Awalan layanan	Tindakan
	memahami medis: StopICD10CMInferenceJob memahami medis: StopPHIDetectionJob memahami medis: StopRxNormInferenceJob memahami medis: StopSNOMEDCTInferenceJob

Awalan layanan	Tindakan
pengoptimal komputasi	pengoptimal komputasi: DeleteRecommendationPreferences
	pengoptimal komputasi: DescribeRecommendationExportJobs
	pengoptimal komputasi: ExportAutoScalingGroupRecommendations
	pengoptimal komputasi: ExportEBSVolumeRecommendations
	pengoptimal komputasi: ExportEC2InstanceRecommendations
	pengoptimal komputasi: ExportECSServiceRecommendations
	pengoptimal komputasi: ExportIdleRecommendations
	pengoptimal komputasi: ExportLambdaFunctionRecommendations
	pengoptimal komputasi: ExportLicenseRecommendations
	pengoptimal komputasi: ExportRDSDatabaseRecommendations
	pengoptimal komputasi: GetEC2RecommendationProjectedMetrics
	pengoptimal komputasi: GetECSServiceRecommendationProjectedMetrics
	pengoptimal komputasi: GetEffectiveRecommendationPreferences
	pengoptimal komputasi: GetEnrollmentStatus
	pengoptimal komputasi: GetEnrollmentStatusesForOrganization
	pengoptimal komputasi: GetRDSDatabaseRecommendationProjectedMetrics
	pengoptimal komputasi: GetRecommendationPreferences
	pengoptimal komputasi: GetRecommendationSummaries
	pengoptimal komputasi: PutRecommendationPreferences

Awalan layanan	Tindakan
	pengoptimal komputasi: UpdateEnrollmentStatus

Awalan layanan	Tindakan
config	konfigurasi: BatchGetResourceConfig konfigurasi: DeleteAggregationAuthorization konfigurasi: DeleteConfigRule konfigurasi: DeleteConfigurationAggregator konfigurasi: DeleteConfigurationRecorder konfigurasi: DeleteConformancePack konfigurasi: DeleteDeliveryChannel konfigurasi: DeleteEvaluationResults konfigurasi: DeleteOrganizationConfigRule konfigurasi: DeleteOrganizationConformancePack konfigurasi: DeletePendingAggregationRequest konfigurasi: DeleteRemediationConfiguration konfigurasi: DeleteRemediationExceptions konfigurasi: DeleteResourceConfig konfigurasi: DeleteRetentionConfiguration konfigurasi: DeleteStoredQuery konfigurasi: DeliverConfigSnapshot konfigurasi: DescribeAggregateComplianceByConfigRules konfigurasi: DescribeAggregateComplianceByConformancePacks konfigurasi: DescribeAggregationAuthorizations konfigurasi: DescribeComplianceByConfigRule

Awalan layanan	Tindakan
	konfigurasi: DescribeComplianceByResource konfigurasi: DescribeConfigRuleEvaluationStatus konfigurasi: DescribeConfigRules konfigurasi: DescribeConfigurationAggregatorSourcesStatus konfigurasi: DescribeConfigurationAggregators konfigurasi: DescribeConfigurationRecorderStatus konfigurasi: DescribeConfigurationRecorders konfigurasi: DescribeConformancePackCompliance konfigurasi: DescribeConformancePackStatus konfigurasi: DescribeConformancePacks konfigurasi: DescribeDeliveryChannelStatus konfigurasi: DescribeDeliveryChannels konfigurasi: DescribeOrganizationConfigRuleStatuses konfigurasi: DescribeOrganizationConfigRules konfigurasi: DescribeOrganizationConformancePackStatuses konfigurasi: DescribeOrganizationConformancePacks konfigurasi: DescribePendingAggregationRequests konfigurasi: DescribeRemediationConfigurations konfigurasi: DescribeRemediationExceptions konfigurasi: DescribeRemediationExecutionStatus konfigurasi: DescribeRetentionConfigurations

Awalan layanan	Tindakan
	konfigurasi: GetComplianceDetailsByConfigRule konfigurasi: GetComplianceDetailsByResource konfigurasi: GetComplianceSummaryByConfigRule konfigurasi: GetComplianceSummaryByResourceType konfigurasi: GetConformancePackComplianceDetails konfigurasi: GetConformancePackComplianceSummary konfigurasi: GetCustomRulePolicy konfigurasi: GetDiscoveredResourceCounts konfigurasi: GetOrganizationConfigRuleDetailedStatus konfigurasi: GetOrganizationConformancePackDetailedStatus konfigurasi: GetOrganizationCustomRulePolicy konfigurasi: GetResourceConfigHistory konfigurasi: GetResourceEvaluationSummary konfigurasi: GetStoredQuery konfigurasi: ListConfigurationRecorders konfigurasi: ListConformancePackComplianceScores konfigurasi: ListConnectors konfigurasi: ListDiscoveredResources konfigurasi: ListResourceEvaluations konfigurasi: ListStoredQueries konfigurasi: PutConfigRule

Awalan layanan	Tindakan
	konfigurasi: PutConfigurationAggregator
	konfigurasi: PutConfigurationRecorder
	konfigurasi: PutConformancePack
	konfigurasi: PutConnector
	konfigurasi: PutDeliveryChannel
	konfigurasi: PutEvaluations
	konfigurasi: PutExternalEvaluation
	konfigurasi: PutOrganizationConfigRule
	konfigurasi: PutOrganizationConformancePack
	konfigurasi: PutRemediationConfigurations
	konfigurasi: PutRemediationExceptions
	konfigurasi: PutResourceConfig
	konfigurasi: PutRetentionConfiguration
	konfigurasi: PutStoredQuery
	konfigurasi: SelectResourceConfig
	konfigurasi: StartConfigRulesEvaluation
	konfigurasi: StartConfigurationRecorder
	konfigurasi: StartRemediationExecution
	konfigurasi: StartResourceEvaluation
	konfigurasi: StopConfigurationRecorder

Awalan layanan	Tindakan
hubungkan	menghubungkan: ActivateEvaluationForm
	menghubungkan: AssociateAnalyticsDataSet
	menghubungkan: AssociateApprovedOrigin
	menghubungkan: AssociateBot
	menghubungkan: AssociateContactWithUser
	menghubungkan: AssociateDefaultVocabulary
	menghubungkan: AssociateEmailAddressAlias
	menghubungkan: AssociateFlow
	menghubungkan: AssociateInstanceStorageConfig
	menghubungkan: AssociateLambdaFunction
	menghubungkan: AssociateLexBot
	menghubungkan: AssociatePhoneNumberContactFlow
	menghubungkan: AssociateQueueEmailAddresses
	menghubungkan: AssociateQueueQuickConnects
	menghubungkan: AssociateRoutingProfileQueues
	menghubungkan: AssociateSecurityKey
	menghubungkan: AssociateUserProficiencies
	menghubungkan: BatchAssociateAnalyticsDataSet
	menghubungkan: BatchCreateDataTableValue
	menghubungkan: BatchDeleteDataTableValue
	menghubungkan: BatchDescribeDataTableValue

Awalan layanan	Tindakan
	menghubungkan: BatchDisassociateAnalyticsDataSet
	menghubungkan: BatchGetFlowAssociation
	menghubungkan: BatchPutContact
	menghubungkan: BatchUpdateDataTableValue
	menghubungkan: ClaimPhoneNumber
	menghubungkan: CreateAgentStatus
	menghubungkan: CreateAttachedFile
	menghubungkan: CreateContact
	menghubungkan: CreateContactFlow
	menghubungkan: CreateContactFlowModule
	menghubungkan: CreateContactFlowModuleAlias
	menghubungkan: CreateContactFlowModuleVersion
	menghubungkan: CreateContactFlowVersion
	menghubungkan: CreateDataTable
	menghubungkan: CreateDataTableAttribute
	menghubungkan: CreateEmailAddress
	menghubungkan: CreateEvaluationForm
	menghubungkan: CreateHoursOfOperation
	menghubungkan: CreateInstance
	menghubungkan: CreateIntegrationAssociation
	menghubungkan: CreateParticipant

Awalan layanan	Tindakan
	menghubungkan: CreatePersistentContactAssociation
	menghubungkan: CreatePredefinedAttribute
	menghubungkan: CreatePrompt
	menghubungkan: CreatePushNotificationRegistration
	menghubungkan: CreateQueue
	menghubungkan: CreateQuickConnect
	menghubungkan: CreateRoutingProfile
	menghubungkan: CreateRule
	menghubungkan: CreateSecurityProfile
	menghubungkan: CreateTaskTemplate
	menghubungkan: CreateTrafficDistributionGroup
	menghubungkan: CreateUseCase
	menghubungkan: CreateUser
	menghubungkan: CreateUserHierarchyGroup
	menghubungkan: CreateView
	menghubungkan: CreateViewVersion
	menghubungkan: CreateVocabulary
	menghubungkan: CreateWorkspace
	menghubungkan: DeactivateEvaluationForm
	menghubungkan: DeleteContactEvaluation
	menghubungkan: DeleteContactFlow

Awalan layanan	Tindakan
	menghubungkan: DeleteContactFlowModule
	menghubungkan: DeleteContactFlowModuleAlias
	menghubungkan: DeleteContactFlowModuleVersion
	menghubungkan: DeleteContactFlowVersion
	menghubungkan: DeleteDataTable
	menghubungkan: DeleteDataTableAttribute
	menghubungkan: DeleteEmailAddress
	menghubungkan: DeleteEvaluationForm
	menghubungkan: DeleteHoursOfOperation
	menghubungkan: DeleteHoursOfOperationOverride
	menghubungkan: DeleteInstance
	menghubungkan: DeleteIntegrationAssociation
	menghubungkan: DeleteNotification
	menghubungkan: DeletePredefinedAttribute
	menghubungkan: DeletePrompt
	menghubungkan: DeletePushNotificationRegistration
	menghubungkan: DeleteQueue
	menghubungkan: DeleteQuickConnect
	menghubungkan: DeleteRoutingProfile
	menghubungkan: DeleteRule
	menghubungkan: DeleteSecurityProfile

Awalan layanan	Tindakan
	menghubungkan: DeleteTaskTemplate
	menghubungkan: DeleteTrafficDistributionGroup
	menghubungkan: DeleteUseCase
	menghubungkan: DeleteUser
	menghubungkan: DeleteUserHierarchyGroup
	menghubungkan: DeleteView
	menghubungkan: DeleteVocabulary
	menghubungkan: DeleteWorkspace
	menghubungkan: DeleteWorkspaceMedia
	menghubungkan: DescribeAttachedFilesConfiguration
	menghubungkan: DescribeAuthenticationProfile
	menghubungkan: DescribeContactFlowModuleAlias
	menghubungkan: DescribeDataTableAttribute
	menghubungkan: DescribeHoursOfOperationOverride
	menghubungkan: DescribeInstanceAttribute
	menghubungkan: DescribeInstanceStorageConfig
	menghubungkan: DescribeNotification
	menghubungkan: DescribePhoneNumber
	menghubungkan: DescribeRule
	menghubungkan: DescribeTrafficDistributionGroup
	menghubungkan: DescribeUserHierarchyStructure

Awalan layanan	Tindakan
	menghubungkan: DescribeVocabulary
	menghubungkan: DisassociateAnalyticsDataSet
	menghubungkan: DisassociateApprovedOrigin
	menghubungkan: DisassociateBot
	menghubungkan: DisassociateEmailAddressAlias
	menghubungkan: DisassociateFlow
	menghubungkan: DisassociateInstanceStorageConfig
	menghubungkan: DisassociateLambdaFunction
	menghubungkan: DisassociateLexBot
	menghubungkan: DisassociatePhoneNumberContactFlow
	menghubungkan: DisassociateQueueEmailAddresses
	menghubungkan: DisassociateQueueQuickConnects
	menghubungkan: DisassociateRoutingProfileQueues
	menghubungkan: DisassociateSecurityKey
	menghubungkan: DisassociateUserProficiencies
	menghubungkan: DismissUserContact
	menghubungkan: EvaluateDataTableValues
	menghubungkan: GetContactAttributes
	menghubungkan: GetContactMetrics
	menghubungkan: GetCurrentMetricData
	menghubungkan: GetCurrentUserData

Awalan layanan	Tindakan
	menghubungkan: GetEffectiveHoursOfOperations
	menghubungkan: GetEvaluationFormValidation
	menghubungkan: GetFederationToken
	menghubungkan: GetFlowAssociation
	menghubungkan: GetMetricData
	menghubungkan: GetMetricDataV2
	menghubungkan: GetPromptFile
	menghubungkan: GetTaskTemplate
	menghubungkan: GetTrafficDistribution
	menghubungkan: ImportPhoneNumber
	menghubungkan: ImportWorkspaceMedia
	menghubungkan: ListAnalyticsDataAssociations
	menghubungkan: ListAnalyticsDataLakeDataSets
	menghubungkan: ListApprovedOrigins
	menghubungkan: ListAssociatedContacts
	menghubungkan: ListAttachedFilesConfigurations
	menghubungkan: ListAuthenticationProfiles
	menghubungkan: ListBots
	menghubungkan: ListContactEvaluations
	menghubungkan: ListContactFlowModuleAliases
	menghubungkan: ListContactFlowModuleVersions

Awalan layanan	Tindakan
	menghubungkan: ListContactFlowModules
	menghubungkan: ListContactFlowVersions
	menghubungkan: ListContactFlows
	menghubungkan: ListContactReferences
	menghubungkan: ListDataTableAttributes
	menghubungkan: ListDataTablePrimaryValues
	menghubungkan: ListDataTableValues
	menghubungkan: ListDataTables
	menghubungkan: ListDefaultVocabularies
	menghubungkan: ListEvaluationFormVersions
	menghubungkan: ListEvaluationForms
	menghubungkan: ListFlowAssociations
	menghubungkan: ListHoursOfOperations
	menghubungkan: ListInstanceAttributes
	menghubungkan: ListInstanceStorageConfigs
	menghubungkan: ListIntegrationAssociations
	menghubungkan: ListLambdaFunctions
	menghubungkan: ListLexBots
	menghubungkan: ListNotifications
	menghubungkan: ListPhoneNumbers
	menghubungkan: ListPhoneNumbersV2

Awalan layanan	Tindakan
	menghubungkan: ListPredefinedAttributes
	menghubungkan: ListPrompts
	menghubungkan: ListQueueEmailAddresses
	menghubungkan: ListQueueQuickConnects
	menghubungkan: ListQueues
	menghubungkan: ListQuickConnects
	menghubungkan: ListRealtimeContactAnalysisSegmentsV2
	menghubungkan: ListRoutingProfileManualAssignmentQueues
	menghubungkan: ListRoutingProfileQueues
	menghubungkan: ListRoutingProfiles
	menghubungkan: ListRules
	menghubungkan: ListSecurityKeys
	menghubungkan: ListSecurityProfileApplications
	menghubungkan: ListSecurityProfileFlowModules
	menghubungkan: ListSecurityProfilePermissions
	menghubungkan: ListSecurityProfiles
	menghubungkan: ListTaskTemplates
	menghubungkan: ListTrafficDistributionGroups
	menghubungkan: ListUseCases
	menghubungkan: ListUserHierarchyGroups
	menghubungkan: ListUserNotifications

Awalan layanan	Tindakan
	menghubungkan: ListUsers
	menghubungkan: ListViewVersions
	menghubungkan: ListViews
	menghubungkan: ListWorkspaceMedia
	menghubungkan: ListWorkspacePages
	menghubungkan: ListWorkspaces
	menghubungkan: MonitorContact
	menghubungkan: PauseContact
	menghubungkan: PutUserStatus
	menghubungkan: ReleasePhoneNumber
	menghubungkan: ReplicateInstance
	menghubungkan: ResumeContact
	menghubungkan: ResumeContactRecording
	menghubungkan: SearchAgentStatuses
	menghubungkan: SearchAvailablePhoneNumbers
	menghubungkan: SearchContactEvaluations
	menghubungkan: SearchContactFlowModules
	menghubungkan: SearchContactFlows
	menghubungkan: SearchContacts
	menghubungkan: SearchDataTables
	menghubungkan: SearchEmailAddresses

Awalan layanan	Tindakan
	menghubungkan: SearchEvaluationForms
	menghubungkan: SearchHoursOfOperations
	menghubungkan: SearchPredefinedAttributes
	menghubungkan: SearchPrompts
	menghubungkan: SearchQueues
	menghubungkan: SearchQuickConnects
	menghubungkan: SearchRoutingProfiles
	menghubungkan: SearchRules
	menghubungkan: SearchSecurityProfiles
	menghubungkan: SearchUserHierarchyGroups
	menghubungkan: SearchViews
	menghubungkan: SearchVocabularies
	menghubungkan: SearchWorkspaceAssociations
	menghubungkan: SearchWorkspaces
	menghubungkan: SendChatIntegrationEvent
	menghubungkan: SendOutboundEmail
	menghubungkan: StartChatContact
	menghubungkan: StartContactConversationalAnalyticsJob
	menghubungkan: StartContactEvaluation
	menghubungkan: StartContactMediaProcessing
	menghubungkan: StartContactRecording

Awalan layanan	Tindakan
	menghubungkan: StartContactStreaming
	menghubungkan: StartEmailContact
	menghubungkan: StartOutboundChatContact
	menghubungkan: StartOutboundEmailContact
	menghubungkan: StartOutboundVoiceContact
	menghubungkan: StartScreenSharing
	menghubungkan: StartTaskContact
	menghubungkan: StartWebRTCContact
	menghubungkan: StopContact
	menghubungkan: StopContactMediaProcessing
	menghubungkan: StopContactRecording
	menghubungkan: StopContactStreaming
	menghubungkan: SubmitContactEvaluation
	menghubungkan: SuspendContactRecording
	menghubungkan: TransferContact
	menghubungkan: UpdateAgentStatus
	menghubungkan: UpdateAttachedFilesConfiguration
	menghubungkan: UpdateAuthenticationProfile
	menghubungkan: UpdateContact
	menghubungkan: UpdateContactAttributes
	menghubungkan: UpdateContactEvaluation

Awalan layanan	Tindakan
	menghubungkan: UpdateContactFlowContent
	menghubungkan: UpdateContactFlowMetadata
	menghubungkan: UpdateContactFlowModuleAlias
	menghubungkan: UpdateContactFlowModuleContent
	menghubungkan: UpdateContactFlowModuleMetadata
	menghubungkan: UpdateContactFlowName
	menghubungkan: UpdateContactRoutingData
	menghubungkan: UpdateContactSchedule
	menghubungkan: UpdateDataTableAttribute
	menghubungkan: UpdateDataTableMetadata
	menghubungkan: UpdateDataTablePrimaryValues
	menghubungkan: UpdateEmailAddressMetadata
	menghubungkan: UpdateEvaluationForm
	menghubungkan: UpdateHoursOfOperation
	menghubungkan: UpdateHoursOfOperationOverride
	menghubungkan: UpdateInstanceAttribute
	menghubungkan: UpdateInstanceStorageConfig
	menghubungkan: UpdateNotificationContent
	menghubungkan: UpdateParticipantAuthentication
	menghubungkan: UpdateParticipantRoleConfig
	menghubungkan: UpdatePhoneNumber

Awalan layanan	Tindakan
	menghubungkan: UpdatePhoneNumberMetadata
	menghubungkan: UpdatePredefinedAttribute
	menghubungkan: UpdatePrompt
	menghubungkan: UpdateQueueHoursOfOperation
	menghubungkan: UpdateQueueMaxContacts
	menghubungkan: UpdateQueueName
	menghubungkan: UpdateQueueOutboundCallerConfig
	menghubungkan: UpdateQueueOutboundEmailConfig
	menghubungkan: UpdateQueueStatus
	menghubungkan: UpdateQuickConnectConfig
	menghubungkan: UpdateQuickConnectName
	menghubungkan: UpdateRoutingProfileAgentAvailabilityTimer
	menghubungkan: UpdateRoutingProfileConcurrency
	menghubungkan: UpdateRoutingProfileDefaultOutboundQueue
	menghubungkan: UpdateRoutingProfileName
	menghubungkan: UpdateRoutingProfileQueues
	menghubungkan: UpdateRule
	menghubungkan: UpdateSecurityProfile
	menghubungkan: UpdateTaskTemplate
	menghubungkan: UpdateTrafficDistribution
	menghubungkan: UpdateUserHierarchy

Awalan layanan	Tindakan
	menghubungkan: UpdateUserHierarchyGroupName menghubungkan: UpdateUserHierarchyStructure menghubungkan: UpdateUserIdentityInfo menghubungkan: UpdateUserNotificationStatus menghubungkan: UpdateUserPhoneConfig menghubungkan: UpdateUserProficiencies menghubungkan: UpdateUserRoutingProfile menghubungkan: UpdateUserSecurityProfiles menghubungkan: UpdateViewContent menghubungkan: UpdateViewMetadata menghubungkan: UpdateWorkspaceMetadata menghubungkan: UpdateWorkspaceTheme menghubungkan: UpdateWorkspaceVisibility
cur	kelebar: DeleteReportDefinition kelebar: DescribeReportDefinitions kelebar: ModifyReportDefinition kelebar: PutReportDefinition

Awalan layanan	Tindakan
databrew	Databrew: BatchDeleteRecipeVersion
	Databrew: CreateDataset
	Databrew: CreateProfileJob
	Databrew: CreateProject
	Databrew: CreateRecipe
	Databrew: CreateRecipeJob
	Databrew: CreateRuleset
	Databrew: CreateSchedule
	Databrew: DeleteDataset
	Databrew: DeleteJob
	Databrew: DeleteProject
	Databrew: DeleteRecipeVersion
	Databrew: DeleteRuleset
	Databrew: DeleteSchedule
	Databrew: DescribeDataset
	Databrew: DescribeJob
	Databrew: DescribeJobRun
	Databrew: DescribeProject
	Databrew: DescribeRecipe
	Databrew: DescribeRuleset
	Databrew: DescribeSchedule

Awalan layanan	Tindakan
	Databrew: ListDatasets
	Databrew: ListJobRuns
	Databrew: ListJobs
	Databrew: ListProjects
	Databrew: ListRecipeVersions
	Databrew: ListRecipes
	Databrew: ListRulesets
	Databrew: ListSchedules
	Databrew: PublishRecipe
	Databrew: SendProjectSessionAction
	Databrew: StartJobRun
	Databrew: StartProjectSession
	Databrew: StopJobRun
	Databrew: UpdateDataset
	Databrew: UpdateProfileJob
	Databrew: UpdateProject
	Databrew: UpdateRecipe
	database: UpdateRecipeJob
	database: UpdateRuleset
	database: UpdateSchedule

Awalan layanan	Tindakan
pertukaran data	pertukaran data: AcceptDataGrant pertukaran data: CancelJob pertukaran data: CreateDataGrant pertukaran data: CreateDataSet pertukaran data: CreateEventAction pertukaran data: CreateJob pertukaran data: CreateRevision pertukaran data: DeleteAsset pertukaran data: DeleteDataGrant pertukaran data: DeleteEventAction pertukaran data: DeleteRevision pertukaran data: GetDataGrant pertukaran data: GetEventAction pertukaran data: GetJob pertukaran data: GetReceivedDataGrant pertukaran data: ListDataGrants pertukaran data: ListDataSetRevisions pertukaran data: ListDataSets pertukaran data: ListEventActions pertukaran data: ListJobs pertukaran data: ListReceivedDataGrants

Awalan layanan	Tindakan
	pertukaran data: ListRevisionAssets pertukaran data: RevokeRevision pertukaran data: SendDataSetNotification pertukaran data: StartJob pertukaran data: UpdateAsset pertukaran data: UpdateDataSet pertukaran data: UpdateEventAction pertukaran data: UpdateRevision

Awalan layanan	Tindakan
datapipeline	datapipeline: ActivatePipeline
	datapipeline: CreatePipeline
	datapipeline: DeactivatePipeline
	datapipeline: DeletePipeline
	datapipeline: DescribeObjects
	datapipeline: DescribePipelines
	datapipeline: EvaluateExpression
	datapipeline: GetPipelineDefinition
	datapipeline: ListPipelines
	datapipeline: PollForTask
	datapipeline: PutPipelineDefinition
	datapipeline: QueryObjects
	datapipeline: ReportTaskProgress
	datapipeline: ReportTaskRunnerHeartbeat
	datapipeline: SetStatus
	datapipeline: SetTaskStatus
	datapipeline: ValidatePipelineDefinition

Awalan layanan	Tindakan
dax	dax: CreateCluster dax: DecreaseReplicationFactor dax: DeleteCluster dax: DeleteParameterGroup dax: DeleteSubnetGroup dax: DescribeClusters dax: DescribeDefaultParameters dax: DescribeEvents dax: DescribeParameterGroups dax: DescribeParameters dax: DescribeSubnetGroups dax: IncreaseReplicationFactor dax: RebootNode dax: UpdateCluster dax: UpdateParameterGroup dax: UpdateSubnetGroup

Awalan layanan	Tindakan
devicefarm	devicefarm: CreateDevicePool
	devicefarm: CreateInstanceProfile
	devicefarm: CreateNetworkProfile
	devicefarm: CreateProject
	devicefarm: CreateRemoteAccessSession
	devicefarm: CreateTestGridProject
	devicefarm: CreateTestGridUrl
	devicefarm: CreateUpload
	DeviceFarm:createVPCEConfiguration
	devicefarm: DeleteDevicePool
	devicefarm: DeleteInstanceProfile
	devicefarm: DeleteNetworkProfile
	devicefarm: DeleteProject
	devicefarm: DeleteRemoteAccessSession
	devicefarm: DeleteRun
	devicefarm: DeleteTestGridProject
	devicefarm: DeleteUpload
	DeviceFarm:deletevpceconfiguration
	devicefarm: GetAccountSettings
	devicefarm: GetDevice
	devicefarm: GetDeviceInstance

Awalan layanan	Tindakan
	devicefarm: GetDevicePool
	devicefarm: GetDevicePoolCompatibility
	devicefarm: GetInstanceProfile
	devicefarm: GetJob
	devicefarm: GetNetworkProfile
	devicefarm: GetOfferingStatus
	devicefarm: GetProject
	devicefarm: GetRemoteAccessSession
	devicefarm: GetRun
	devicefarm: GetSuite
	devicefarm: GetTest
	devicefarm: GetTestGridProject
	devicefarm: GetTestGridSession
	devicefarm: GetUpload
	DeviceFarm:GetVPCEConfiguration
	devicefarm: ListArtifacts
	devicefarm: ListDeviceInstances
	devicefarm: ListDevicePools
	devicefarm: ListDevices
	devicefarm: ListInstanceProfiles
	devicefarm: ListJobs

Awalan layanan	Tindakan
	devicefarm: ListNetworkProfiles
	devicefarm: ListOfferingPromotions
	devicefarm: ListOfferingTransactions
	devicefarm: ListOfferings
	devicefarm: ListProjects
	devicefarm: ListRemoteAccessSessions
	devicefarm: ListRuns
	devicefarm: ListSamples
	devicefarm: ListSuites
	devicefarm: ListTestGridProjects
	devicefarm: ListTestGridSessionActions
	devicefarm: ListTestGridSessionArtifacts
	devicefarm: ListTestGridSessions
	devicefarm: ListTests
	devicefarm: ListUniqueProblems
	devicefarm: ListUploads
	DeviceFarm:ListVPCEConfigurations
	devicefarm: PurchaseOffering
	devicefarm: RenewOffering
	devicefarm: ScheduleRun
	devicefarm: StopJob

Awalan layanan	Tindakan
	devicefarm: StopRemoteAccessSession
	devicefarm: StopRun
	devicefarm: UpdateDeviceInstance
	devicefarm: UpdateDevicePool
	devicefarm: UpdateInstanceProfile
	devicefarm: UpdateNetworkProfile
	devicefarm: UpdateProject
	devicefarm: UpdateTestGridProject
	devicefarm: UpdateUpload
	DeviceFarm:updatevpceconfiguration

Awalan layanan	Tindakan
devops-guru	devops-guru: AddNotificationChannel devops-guru: DeleteInsight devops-guru: DescribeAccountHealth devops-guru: DescribeAccountOverview devops-guru: DescribeAnomaly devops-guru: DescribeEventSourcesConfig devops-guru: DescribeFeedback devops-guru: DescribeInsight devops-guru: DescribeOrganizationHealth devops-guru: DescribeOrganizationOverview devops-guru: DescribeOrganizationResourceCollectionHealth devops-guru: DescribeResourceCollectionHealth devops-guru: DescribeServiceIntegration devops-guru: GetCostEstimation devops-guru: GetResourceCollection devops-guru: ListAnomaliesForInsight devops-guru: ListAnomalousLogGroups devops-guru: ListEvents devops-guru: ListInsights devops-guru: ListMonitoredResources devops-guru: ListNotificationChannels

Awalan layanan	Tindakan
	devops-guru: ListOrganizationInsights devops-guru: ListRecommendations devops-guru: PutFeedback devops-guru: RemoveNotificationChannel devops-guru: SearchInsights devops-guru: SearchOrganizationInsights devops-guru: StartCostEstimation devops-guru: UpdateEventSourcesConfig devops-guru: UpdateResourceCollection devops-guru: UpdateServiceIntegration

Awalan layanan	Tindakan
DirectConnect	directconnect: AcceptDirectConnectGatewayAssociationProposal
	directconnect: AllocateConnectionOnInterconnect
	directconnect: AllocateHostedConnection
	directconnect: AllocatePrivateVirtualInterface
	directconnect: AllocatePublicVirtualInterface
	directconnect: AllocateTransitVirtualInterface
	directconnect: AssociateConnectionWithLag
	directconnect: AssociateHostedConnection
	directconnect: AssociateMacSecKey
	directconnect: AssociateVirtualInterface
	directconnect: ConfirmConnection
	directconnect: ConfirmCustomerAgreement
	directconnect: ConfirmPrivateVirtualInterface
	directconnect: ConfirmPublicVirtualInterface
	directconnect: ConfirmTransitVirtualInterface
	DirectConnect:CreateBGPpeer
	directconnect: CreateConnection
	directconnect: CreateDirectConnectGateway
	directconnect: CreateDirectConnectGatewayAssociation
	directconnect: CreateDirectConnectGatewayAssociationProposal
	directconnect: CreateInterconnect

Awalan layanan	Tindakan
	directconnect: CreateLag
	directconnect: CreatePrivateVirtualInterface
	directconnect: CreatePublicVirtualInterface
	directconnect: CreateTransitVirtualInterface
	DirectConnect:DeleteBGPPeer
	directconnect: DeleteConnection
	directconnect: DeleteDirectConnectGateway
	directconnect: DeleteDirectConnectGatewayAssociation
	directconnect: DeleteDirectConnectGatewayAssociationProposal
	directconnect: DeleteInterconnect
	directconnect: DeleteLag
	directconnect: DeleteVirtualInterface
	directconnect: DescribeConnectionLoa
	directconnect: DescribeConnections
	directconnect: DescribeConnectionsOnInterconnect
	directconnect: DescribeCustomerMetadata
	directconnect: DescribeDirectConnectGatewayAssociationProposals
	directconnect: DescribeDirectConnectGatewayAssociations
	directconnect: DescribeDirectConnectGatewayAttachments
	directconnect: DescribeDirectConnectGateways

Awalan layanan	Tindakan
	directconnect: DescribeHostedConnections
	directconnect: DescribeInterconnectLoa
	directconnect: DescribeInterconnects
	directconnect: DescribeLags
	directconnect: DescribeLoa
	directconnect: DescribeLocations
	directconnect: DescribeRouterConfiguration
	directconnect: DescribeVirtualGateways
	directconnect: DescribeVirtualInterfaces
	directconnect: DisassociateConnectionFromLag
	directconnect: DisassociateMacSecKey
	directconnect: ListVirtualInterfaceTestHistory
	directconnect: StartBgpFailoverTest
	directconnect: StopBgpFailoverTest
	directconnect: UpdateConnection
	directconnect: UpdateDirectConnectGateway
	directconnect: UpdateDirectConnectGatewayAssociation
	directconnect: UpdateLag
	directconnect: UpdateVirtualInterfaceAttributes

Awalan layanan	Tindakan
dlm	dlm: CreateLifecyclePolicy dlm: DeleteLifecyclePolicy dlm: GetLifecyclePolicies dlm: GetLifecyclePolicy dlm: UpdateLifecyclePolicy

Awalan layanan	Tindakan
dms	dms: ApplyPendingMaintenanceAction
	dms: AssociateExtensionPack
	dms: BatchStartRecommendations
	dms: CancelMetadataModelCreation
	dms: CancelReplicationTaskAssessmentRun
	dms: CreateDataProvider
	dms: CreateEndpoint
	dms: CreateEventSubscription
	dms: CreateInstanceProfile
	dms: CreateMigrationProject
	dms: CreateReplicationConfig
	dms: CreateReplicationInstance
	dms: CreateReplicationSubnetGroup
	dms: CreateReplicationTask
	dms: DeleteCertificate
	dms: DeleteConnection
	dms: DeleteDataMigration
	dms: DeleteDataProvider
	dms: DeleteEndpoint
	dms: DeleteEventSubscription
	dms: DeleteFleetAdvisorCollector

Awalan layanan	Tindakan
	dms: DeleteFleetAdvisorDatabases
	dms: DeleteInstanceProfile
	dms: DeleteMigrationProject
	dms: DeleteReplicationConfig
	dms: DeleteReplicationInstance
	dms: DeleteReplicationSubnetGroup
	dms: DeleteReplicationTask
	dms: DeleteReplicationTaskAssessmentRun
	dms: DescribeAccountAttributes
	dms: DescribeApplicableIndividualAssessments
	dms: DescribeCertificates
	dms: DescribeConnections
	dms: DescribeDataMigrations
	dms: DescribeEndpointSettings
	dms: DescribeEndpointTypes
	dms: DescribeEndpoints
	dms: DescribeEngineVersions
	dms: DescribeEventCategories
	dms: DescribeEventSubscriptions
	dms: DescribeEvents
	dms: DescribeFleetAdvisorCollectors

Awalan layanan	Tindakan
	dms: DescribeFleetAdvisorDatabases
	dms: DescribeFleetAdvisorLsaAnalysis
	dms: DescribeFleetAdvisorSchemaObjectSummary
	dms: DescribeFleetAdvisorSchemas
	dms: DescribeMetadataModel
	dms: DescribeMetadataModelChildren
	dms: DescribeMetadataModelCreations
	dms: DescribeMetadataModelImports
	dms: DescribeOrderableReplicationInstances
	dms: DescribePendingMaintenanceActions
	dms: DescribeRecommendationLimitations
	dms: DescribeRecommendations
	dms: DescribeRefreshSchemasStatus
	dms: DescribeReplicationConfigs
	dms: DescribeReplicationInstanceTaskLogs
	dms: DescribeReplicationInstances
	dms: DescribeReplicationSubnetGroups
	dms: DescribeReplicationTableStatistics
	dms: DescribeReplicationTaskAssessmentResults
	dms: DescribeReplicationTaskAssessmentRuns
	dms: DescribeReplicationTaskIndividualAssessments

Awalan layanan	Tindakan
	dms: DescribeReplicationTasks
	dms: DescribeReplications
	dms: DescribeSchemas
	dms: DescribeTableStatistics
	dms: ExportMetadataModelAssessment
	dms: ImportCertificate
	dms: ListDataProviders
	dms: ListExtensionPacks
	dms: ListInstanceProfiles
	dms: ListMetadataModelAssessments
	dms: ListMetadataModelConversions
	dms: ListMetadataModelExports
	dms: ListMigrationProjects
	dms: ModifyDataMigration
	dms: ModifyEndpoint
	dms: ModifyEventSubscription
	dms: ModifyReplicationConfig
	dms: ModifyReplicationInstance
	dms: ModifyReplicationSubnetGroup
	dms: ModifyReplicationTask
	dms: MoveReplicationTask

Awalan layanan	Tindakan
	dms: RebootReplicationInstance
	dms: RefreshSchemas
	dms: ReloadReplicationTables
	dms: ReloadTables
	dms: RunFleetAdvisorLsaAnalysis
	dms: StartMetadataModelAssessment
	dms: StartMetadataModelConversion
	dms: StartMetadataModelCreation
	dms: StartMetadataModelExportAsScripts
	dms: StartMetadataModelExportToTarget
	dms: StartRecommendations
	dms: StartReplication
	dms: StartReplicationTask
	dms: StartReplicationTaskAssessment
	dms: StopDataMigration
	dms: StopReplicationTask
	dms: TestConnection
	dms: UpdateConversionConfiguration
	dms: UpdateDataProvider
	dms: UpdateInstanceProfile
	dms: UpdateMigrationProject

Awalan layanan	Tindakan
	dms: UpdateSubscriptionsToEventBridge
docdb-elastis	docdb-elastis: ApplyPendingMaintenanceAction docdb-elastis: CopyClusterSnapshot docdb-elastis: DeleteCluster docdb-elastis: DeleteClusterSnapshot docdb-elastis: GetCluster docdb-elastis: GetClusterSnapshot docdb-elastis: GetPendingMaintenanceAction docdb-elastis: ListClusterSnapshots docdb-elastis: ListClusters docdb-elastis: ListPendingMaintenanceActions docdb-elastis: RestoreClusterFromSnapshot docdb-elastis: StartCluster docdb-elastis: StopCluster docdb-elastis: UpdateCluster

Awalan layanan	Tindakan
dynamodb	dinamodb: CreateBackup
	dinamodb: CreateGlobalTable
	dinamodb: CreateTable
	dinamodb: DeleteBackup
	dinamodb: DeleteTable
	dinamodb: DescribeBackup
	dinamodb: DescribeContinuousBackups
	dinamodb: DescribeContributorInsights
	dinamodb: DescribeEndpoints
	dinamodb: DescribeExport
	dinamodb: DescribeGlobalTable
	dinamodb: DescribeGlobalTableSettings
	dinamodb: DescribeImport
	dinamodb: DescribeKinesisStreamingDestination
	dinamodb: DescribeLimits
	dinamodb: DescribeStream
	dinamodb: DescribeTable
	dinamodb: DescribeTableReplicaAutoScaling
	dinamodb: DescribeTimeToLive
	dinamodb: DisableKinesisStreamingDestination
	dinamodb: EnableKinesisStreamingDestination

Awalan layanan	Tindakan
	dinamodb: ExportTableToPointInTime
	dinamodb: GetResourcePolicy
	dinamodb: ImportTable
	dinamodb: ListBackups
	dinamodb: ListContributorInsights
	dinamodb: ListExports
	dinamodb: ListGlobalTables
	dinamodb: ListImports
	dinamodb: ListStreams
	dinamodb: ListTables
	dinamodb: RestoreTableFromBackup
	dinamodb: RestoreTableToPointInTime
	dinamodb: UpdateContinuousBackups
	dinamodb: UpdateContributorInsights
	dinamodb: UpdateGlobalTable
	dinamodb: UpdateGlobalTableSettings
	dinamodb: UpdateKinesisStreamingDestination
	dinamodb: UpdateTable
	dinamodb: UpdateTableReplicaAutoScaling
	dinamodb: UpdateTimeToLive

Awalan layanan	Tindakan
ebs	ebs: CompleteSnapshot ebs: StartSnapshot

Awalan layanan	Tindakan
ec2	EC2: AcceptAddressTransfer
	EC2: AcceptCapacityReservationBillingOwnership
	EC2: AcceptReservedInstancesExchangeQuote
	EC2: AcceptTransitGatewayClientVpnAttachment
	EC2: AcceptTransitGatewayMulticastDomainAssociations
	EC2: AcceptTransitGatewayPeeringAttachment
	EC2: AcceptTransitGatewayVpcAttachment
	EC2: AcceptVpcEndpointConnections
	EC2: AcceptVpcPeeringConnection
	EC2: AdvertiseByoipCidr
	EC2: AllocateAddress
	EC2: AllocateHosts
	EC2: AllocateIpamPoolCidr
	EC2: ApplySecurityGroupsToClientVpnTargetNetwork
	EC2: AssignIpv6Addresses
	EC2: AssignPrivateIpAddresses
	EC2: AssignPrivateNatGatewayAddress
	EC2: AssociateAddress
	EC2: AssociateCapacityReservationBillingOwner
	EC2: AssociateClientVpnTargetNetwork
	EC2: AssociateDhcpOptions

Awalan layanan	Tindakan
	EC2: AssociateEnclaveCertificateIamRole
	EC2: AssociateIamInstanceProfile
	EC2: AssociateInstanceEventWindow
	EC2: AssociateIpamByoasn
	EC2: AssociateIpamResourceDiscovery
	EC2: AssociateNatGatewayAddress
	EC2: AssociateRouteServer
	EC2: AssociateRouteTable
	EC2: AssociateSecurityGroupVpc
	EC2: AssociateSubnetCidrBlock
	EC2: AssociateTransitGatewayMulticastDomain
	EC2: AssociateTransitGatewayPolicyTable
	EC2: AssociateTransitGatewayRouteTable
	EC2: AssociateTrunkInterface
	EC2: AssociateVpcCidrBlock
	EC2: AttachClassicLinkVpc
	EC2: AttachImageWatermark
	EC2: AttachInternetGateway
	EC2: AttachNetworkInterface
	EC2: AttachVerifiedAccessTrustProvider
	EC2: AttachVolume

Awalan layanan	Tindakan
	EC2: AttachVpnGateway
	EC2: AuthorizeClientVpnIngress
	EC2: AuthorizeSecurityGroupEgress
	EC2: AuthorizeSecurityGroupIngress
	EC2: BundleInstance
	EC2: CancelBundleTask
	EC2: CancelCapacityReservation
	EC2: CancelCapacityReservationFleets
	EC2: CancelConversionTask
	EC2: CancelDeclarativePoliciesReport
	EC2: CancelExportTask
	EC2: CancelImageLaunchPermission
	EC2: CancelImportTask
	EC2: CancelReservedInstancesListing
	EC2: CancelSpotFleetRequests
	EC2: CancelSpotInstanceRequests
	EC2: ConfirmProductInstance
	EC2: CopyFpgaImage
	EC2: CopyImage
	EC2: CopySnapshot
	EC2: CopyVolumes

Awalan layanan	Tindakan
	EC2: CreateCapacityManagerDataExport
	EC2: CreateCapacityReservation
	EC2: CreateCapacityReservationBySplitting
	EC2: CreateCapacityReservationCancellationQuote
	EC2: CreateCapacityReservationFleet
	EC2: CreateCarrierGateway
	EC2: CreateClientVpnEndpoint
	EC2: CreateClientVpnRoute
	EC2: CreateCoipCidr
	EC2: CreateCoipPool
	EC2: CreateCustomerGateway
	EC2: CreateDefaultSubnet
	EC2: CreateDefaultVpc
	EC2: CreateDelegateMacVolumeOwnershipTask
	EC2: CreateDhcpOptions
	EC2: CreateEgressOnlyInternetGateway
	EC2: CreateFleet
	EC2: CreateFlowLogs
	EC2: CreateFpgaImage
	EC2: CreateImage
	EC2: CreateImageUsageReport

Awalan layanan	Tindakan
	EC2: CreateInstanceConnectEndpoint
	EC2: CreateInstanceEventWindow
	EC2: CreateInstanceExportTask
	EC2: CreateInternetGateway
	EC2: CreateInterruptibleCapacityReservationAllocation
	EC2: CreateIpam
	EC2: CreateIpamExternalResourceVerificationToken
	EC2: CreateIpamPolicy
	EC2: CreateIpamPool
	EC2: CreateIpamPrefixListResolver
	EC2: CreateIpamPrefixListResolverTarget
	EC2: CreateIpamResourceDiscovery
	EC2: CreateIpamScope
	EC2: CreateKeyPair
	EC2: CreateLaunchTemplateVersion
	EC2: CreateLocalGatewayRoute
	EC2: CreateLocalGatewayRouteTable
	EC2: CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2: CreateLocalGatewayRouteTableVpcAssociation
	EC2: CreateLocalGatewayVirtualInterface

Awalan layanan	Tindakan
	EC2: CreateLocalGatewayVirtualInterfaceGroup
	EC2: CreateMacSystemIntegrityProtectionModificationTask
	EC2: CreateManagedPrefixList
	EC2: CreateNatGateway
	EC2: CreateNetworkAcl
	EC2: CreateNetworkAclEntry
	EC2: CreateNetworkInsightsAccessScope
	EC2: CreateNetworkInsightsPath
	EC2: CreateNetworkInterface
	EC2: CreateNetworkInterfacePermission
	EC2: CreatePlacementGroup
	EC2: CreatePublicIpv4Pool
	EC2: CreateReplaceRootVolumeTask
	EC2: CreateReservedInstancesListing
	EC2: CreateRestoreImageTask
	EC2: CreateRoute
	EC2: CreateRouteServer
	EC2: CreateRouteServerEndpoint
	EC2: CreateRouteServerPeer
	EC2: CreateRouteTable
	EC2: CreateSecondaryNetwork

Awalan layanan	Tindakan
	EC2: CreateSecondarySubnet
	EC2: CreateSecurityGroup
	EC2: CreateSnapshots
	EC2: CreateSpotDatafeedSubscription
	EC2: CreateStoreImageTask
	EC2: CreateSubnet
	EC2: CreateSubnetCidrReservation
	EC2: CreateTrafficMirrorFilter
	EC2: CreateTrafficMirrorFilterRule
	EC2: CreateTrafficMirrorSession
	EC2: CreateTrafficMirrorTarget
	EC2: CreateTransitGateway
	EC2: CreateTransitGatewayConnect
	EC2: CreateTransitGatewayConnectPeer
	EC2: CreateTransitGatewayMeteringPolicy
	EC2: CreateTransitGatewayMeteringPolicyEntry
	EC2: CreateTransitGatewayMulticastDomain
	EC2: CreateTransitGatewayPeeringAttachment
	EC2: CreateTransitGatewayPolicyTable
	EC2: CreateTransitGatewayPrefixListReference
	EC2: CreateTransitGatewayRoute

Awalan layanan	Tindakan
	EC2: CreateTransitGatewayRouteTable
	EC2: CreateTransitGatewayRouteTableAnnouncement
	EC2: CreateTransitGatewayVpcAttachment
	EC2: CreateVerifiedAccessEndpoint
	EC2: CreateVerifiedAccessGroup
	EC2: CreateVerifiedAccessInstance
	EC2: CreateVerifiedAccessTrustProvider
	EC2: CreateVolume
	EC2: CreateVpc
	EC2: CreateVpcBlockPublicAccessExclusion
	EC2: CreateVpcEncryptionControl
	EC2: CreateVpcEndpoint
	EC2: CreateVpcEndpointConnectionNotification
	EC2: CreateVpcEndpointServiceConfiguration
	EC2: CreateVpcPeeringConnection
	EC2: CreateVpnConcentrator
	EC2: CreateVpnConnection
	EC2: CreateVpnConnectionRoute
	EC2: CreateVpnGateway
	EC2: DeleteCapacityManagerDataExport
	EC2: DeleteCarrierGateway

Awalan layanan	Tindakan
	EC2: DeleteClientVpnEndpoint
	EC2: DeleteClientVpnRoute
	EC2: DeleteCoipCidr
	EC2: DeleteCoipPool
	EC2: DeleteCustomerGateway
	EC2: DeleteDhcpOptions
	EC2: DeleteEgressOnlyInternetGateway
	EC2: DeleteFleets
	EC2: DeleteFlowLogs
	EC2: DeleteFpgaImage
	EC2: DeleteImageUsageReport
	EC2: DeleteInstanceConnectEndpoint
	EC2: DeleteInstanceEventWindow
	EC2: DeleteInternetGateway
	EC2: DeleteIpam
	EC2: DeleteIpamExternalResourceVerificationToken
	EC2: DeleteIpamPolicy
	EC2: DeleteIpamPool
	EC2: DeleteIpamPrefixListResolver
	EC2: DeleteIpamPrefixListResolverTarget
	EC2: DeleteIpamResourceDiscovery

Awalan layanan	Tindakan
	EC2: DeleteIpamScope
	EC2: DeleteKeyPair
	EC2: DeleteLaunchTemplate
	EC2: DeleteLaunchTemplateVersions
	EC2: DeleteLocalGatewayRoute
	EC2: DeleteLocalGatewayRouteTable
	EC2: DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2: DeleteLocalGatewayRouteTableVpcAssociation
	EC2: DeleteLocalGatewayVirtualInterface
	EC2: DeleteLocalGatewayVirtualInterfaceGroup
	EC2: DeleteManagedPrefixList
	EC2: DeleteNatGateway
	EC2: DeleteNetworkAcl
	EC2: DeleteNetworkAclEntry
	EC2: DeleteNetworkInsightsAccessScope
	EC2: DeleteNetworkInsightsAccessScopeAnalysis
	EC2: DeleteNetworkInsightsAnalysis
	EC2: DeleteNetworkInsightsPath
	EC2: DeleteNetworkInterface
	EC2: DeleteNetworkInterfacePermission

Awalan layanan	Tindakan
	EC2: DeletePlacementGroup
	EC2: DeletePublicIpv4Pool
	EC2: DeleteQueuedReservedInstances
	EC2: DeleteRoute
	EC2: DeleteRouteServer
	EC2: DeleteRouteServerEndpoint
	EC2: DeleteRouteServerPeer
	EC2: DeleteRouteTable
	EC2: DeleteSecondaryNetwork
	EC2: DeleteSecondarySubnet
	EC2: DeleteSecurityGroup
	EC2: DeleteSpotDatafeedSubscription
	EC2: DeleteSubnet
	EC2: DeleteSubnetCidrReservation
	EC2: DeleteTrafficMirrorFilter
	EC2: DeleteTrafficMirrorFilterRule
	EC2: DeleteTrafficMirrorSession
	EC2: DeleteTrafficMirrorTarget
	EC2: DeleteTransitGateway
	EC2: DeleteTransitGatewayClientVpnAttachment
	EC2: DeleteTransitGatewayConnect

Awalan layanan	Tindakan
	EC2: DeleteTransitGatewayConnectPeer
	EC2: DeleteTransitGatewayMeteringPolicy
	EC2: DeleteTransitGatewayMeteringPolicyEntry
	EC2: DeleteTransitGatewayMulticastDomain
	EC2: DeleteTransitGatewayPeeringAttachment
	EC2: DeleteTransitGatewayPolicyTable
	EC2: DeleteTransitGatewayPrefixListReference
	EC2: DeleteTransitGatewayRoute
	EC2: DeleteTransitGatewayRouteTable
	EC2: DeleteTransitGatewayRouteTableAnnouncement
	EC2: DeleteTransitGatewayVpcAttachment
	EC2: DeleteVerifiedAccessEndpoint
	EC2: DeleteVerifiedAccessGroup
	EC2: DeleteVerifiedAccessInstance
	EC2: DeleteVerifiedAccessTrustProvider
	EC2: DeleteVolume
	EC2: DeleteVpc
	EC2: DeleteVpcBlockPublicAccessExclusion
	EC2: DeleteVpcEncryptionControl
	EC2: DeleteVpcEndpointConnectionNotifications
	EC2: DeleteVpcEndpointServiceConfigurations

Awalan layanan	Tindakan
	EC2: DeleteVpcEndpoints
	EC2: DeleteVpcPeeringConnection
	EC2: DeleteVpnConcentrator
	EC2: DeleteVpnConnection
	EC2: DeleteVpnConnectionRoute
	EC2: DeleteVpnGateway
	EC2: DeprovisionByoipCidr
	EC2: DeprovisionIpamByoasn
	EC2: DeprovisionIpamPoolCidr
	EC2: DeprovisionPublicIpv4PoolCidr
	EC2: DeregisterImage
	EC2: DeregisterInstanceEventNotificationAttributes
	EC2: DeregisterTransitGatewayMulticastGroupMembers
	EC2: DeregisterTransitGatewayMulticastGroupSources
	EC2: DescribeAccountAttributes
	EC2: DescribeAddressTransfers
	EC2: DescribeAddresses
	EC2: DescribeAddressesAttribute
	EC2: DescribeAggregateIdFormat
	EC2: DescribeAvailabilityZones
	EC2: DescribeAwsNetworkPerformanceMetricSubscriptions

Awalan layanan	Tindakan
	EC2: DescribeBundleTasks
	EC2: DescribeByoipCidrs
	EC2: DescribeCapacityBlockExtensionHistory
	EC2: DescribeCapacityBlockExtensionOfferings
	EC2: DescribeCapacityBlockStatus
	EC2: DescribeCapacityBlocks
	EC2: DescribeCapacityManagerDataExports
	EC2: DescribeCapacityReservationBillingRequests
	EC2: DescribeCapacityReservationCancellationQuotes
	EC2: DescribeCapacityReservationFleets
	EC2: DescribeCapacityReservationTopology
	EC2: DescribeCapacityReservations
	EC2: DescribeCarrierGateways
	EC2: DescribeClassicLinkInstances
	EC2: DescribeClientVpnAuthorizationRules
	EC2: DescribeClientVpnConnections
	EC2: DescribeClientVpnEndpoints
	EC2: DescribeClientVpnRoutes
	EC2: DescribeClientVpnTargetNetworks
	EC2: DescribeCoipPools
	EC2: DescribeConversionTasks

Awalan layanan	Tindakan
	EC2: DescribeCustomerGateways
	EC2: DescribeDeclarativePoliciesReports
	EC2: DescribeDhcpOptions
	EC2: DescribeEgressOnlyInternetGateways
	EC2: DescribeElasticGpus
	EC2: DescribeExportImageTasks
	EC2: DescribeExportTasks
	EC2: DescribeFastLaunchImages
	EC2: DescribeFastSnapshotRestores
	EC2: DescribeFleetHistory
	EC2: DescribeFleetInstances
	EC2: DescribeFleets
	EC2: DescribeFlowLogs
	EC2: DescribeFpgaImageAttribute
	EC2: DescribeFpgaImages
	EC2: DescribeHostReservationOfferings
	EC2: DescribeHostReservations
	EC2: DescribeHosts
	EC2: DescribeIamInstanceProfileAssociations
	EC2: DescribeIdFormat
	EC2: DescribeIdentityIdFormat

Awalan layanan	Tindakan
	EC2: DescribeImageAttribute
	EC2: DescribeImageReferences
	EC2: DescribeImageUsageReportEntries
	EC2: DescribeImageUsageReports
	EC2: DescribeImportImageTasks
	EC2: DescribeImportSnapshotTasks
	EC2: DescribeInstanceConnectEndpoints
	EC2: DescribeInstanceCreditSpecifications
	EC2: DescribeInstanceEventNotificationAttributes
	EC2: DescribeInstanceEventWindows
	EC2: DescribeInstanceImageMetadata
	EC2: DescribeInstanceSqlHaHistoryStates
	EC2: DescribeInstanceSqlHaStates
	EC2: DescribeInstanceTopology
	EC2: DescribeInstanceTypes
	EC2: DescribeInternetGateways
	EC2: DescribeIpamByoasn
	EC2: DescribeIpamExternalResourceVerificationTokens
	EC2: DescribeIpamPolicies
	EC2: DescribeIpamPoolAllocations
	EC2: DescribeIpamPools

Awalan layanan	Tindakan
	EC2: DescribeIpamPrefixListResolverTargets
	EC2: DescribeIpamPrefixListResolvers
	EC2: DescribeIpamResourceDiscoveries
	EC2: DescribeIpamResourceDiscoveryAssociations
	EC2: DescribeIpamScopes
	EC2: DescribeIpams
	EC2: DescribeIpv6Pools
	EC2: DescribeKeyPairs
	EC2: DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	EC2: DescribeLocalGatewayRouteTableVpcAssociations
	EC2: DescribeLocalGatewayRouteTables
	EC2: DescribeLocalGatewayVirtualInterfaceGroups
	EC2: DescribeLocalGatewayVirtualInterfaces
	EC2: DescribeLocalGateways
	EC2: DescribeLockedSnapshots
	EC2: DescribeMacHosts
	EC2: DescribeMacModificationTasks
	EC2: DescribeManagedPrefixLists
	EC2: DescribeMovingAddresses
	EC2: DescribeNatGateways

Awalan layanan	Tindakan
	EC2: DescribeNetworkAcls
	EC2: DescribeNetworkInsightsAccessScopeAnalyses
	EC2: DescribeNetworkInsightsAccessScopes
	EC2: DescribeNetworkInsightsAnalyses
	EC2: DescribeNetworkInsightsPaths
	EC2: DescribeNetworkInterfaceAttribute
	EC2: DescribeNetworkInterfacePermissions
	EC2: DescribeNetworkInterfaces
	EC2: DescribeOutpostLags
	EC2: DescribePlacementGroups
	EC2: DescribePrefixLists
	EC2: DescribePrincipalIdFormat
	EC2: DescribePublicIpv4Pools
	EC2: DescribeRegions
	EC2: DescribeReplaceRootVolumeTasks
	EC2: DescribeReservedInstances
	EC2: DescribeReservedInstancesListings
	EC2: DescribeReservedInstancesModifications
	EC2: DescribeReservedInstancesOfferings
	EC2: DescribeRouteServerEndpoints
	EC2: DescribeRouteServerPeers

Awalan layanan	Tindakan
	EC2: DescribeRouteServers
	EC2: DescribeRouteTables
	EC2: DescribeScheduledInstanceAvailability
	EC2: DescribeScheduledInstances
	EC2: DescribeSecondaryInterfaces
	EC2: DescribeSecondaryNetworks
	EC2: DescribeSecondarySubnets
	EC2: DescribeSecurityGroupReferences
	EC2: DescribeSecurityGroupRules
	EC2: DescribeSecurityGroupVpcAssociations
	EC2: DescribeSecurityGroups
	EC2: DescribeServiceLinkVirtualInterfaces
	EC2: DescribeSnapshotAttribute
	EC2: DescribeSnapshotTierStatus
	EC2: DescribeSpotDatafeedSubscription
	EC2: DescribeSpotFleetInstances
	EC2: DescribeSpotFleetRequestHistory
	EC2: DescribeSpotFleetRequests
	EC2: DescribeSpotInstanceRequests
	EC2: DescribeSpotPriceHistory
	EC2: DescribeStaleSecurityGroups

Awalan layanan	Tindakan
	EC2: DescribeStoreImageTasks
	EC2: DescribeTrafficMirrorFilterRules
	EC2: DescribeTrafficMirrorFilters
	EC2: DescribeTrafficMirrorSessions
	EC2: DescribeTrafficMirrorTargets
	EC2: DescribeTransitGatewayAttachments
	EC2: DescribeTransitGatewayConnectPeers
	EC2: DescribeTransitGatewayConnects
	EC2: DescribeTransitGatewayMeteringPolicies
	EC2: DescribeTransitGatewayMulticastDomains
	EC2: DescribeTransitGatewayPeeringAttachments
	EC2: DescribeTransitGatewayPolicyTables
	EC2: DescribeTransitGatewayRouteTableAnnouncements
	EC2: DescribeTransitGatewayRouteTables
	EC2: DescribeTransitGatewayVpcAttachments
	EC2: DescribeTransitGateways
	EC2: DescribeTrunkInterfaceAssociations
	EC2: DescribeVerifiedAccessEndpoints
	EC2: DescribeVerifiedAccessGroups
	EC2: DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2: DescribeVerifiedAccessInstances

Awalan layanan	Tindakan
	EC2: DescribeVerifiedAccessTrustProviders
	EC2: DescribeVolumeAttribute
	EC2: DescribeVolumeStatus
	EC2: DescribeVolumes
	EC2: DescribeVolumesModifications
	EC2: DescribeVpcAttribute
	EC2: DescribeVpcBlockPublicAccessExclusions
	EC2: DescribeVpcBlockPublicAccessOptions
	EC2: DescribeVpcClassicLink
	EC2: DescribeVpcClassicLinkDnsSupport
	EC2: DescribeVpcEncryptionControls
	EC2: DescribeVpcEndpointAssociations
	EC2: DescribeVpcEndpointConnectionNotifications
	EC2: DescribeVpcEndpointConnections
	EC2: DescribeVpcEndpointServiceConfigurations
	EC2: DescribeVpcEndpointServicePermissions
	EC2: DescribeVpcEndpointServices
	EC2: DescribeVpcEndpoints
	EC2: DescribeVpcPeeringConnections
	EC2: DescribeVpcs
	EC2: DescribeVpnConcentrators

Awalan layanan	Tindakan
	EC2: DescribeVpnConnections
	EC2: DescribeVpnGateways
	EC2: DetachClassicLinkVpc
	EC2: DetachImageWatermark
	EC2: DetachInternetGateway
	EC2: DetachNetworkInterface
	EC2: DetachVerifiedAccessTrustProvider
	EC2: DetachVolume
	EC2: DetachVpnGateway
	EC2: DisableAddressTransfer
	EC2: DisableAllowedImagesSettings
	EC2: DisableAwsNetworkPerformanceMetricSubscription
	EC2: DisableCapacityManager
	EC2: DisableEbsEncryptionByDefault
	EC2: DisableFastLaunch
	EC2: DisableFastSnapshotRestores
	EC2: DisableImage
	EC2: DisableImageBlockPublicAccess
	EC2: DisableImageDeprecation
	EC2: DisableImageDeregistrationProtection
	EC2: DisableInstanceSqlHaStandbyDetections

Awalan layanan	Tindakan
	EC2: DisableIpamOrganizationAdminAccount
	EC2: DisableIpamPolicy
	EC2: DisableRouteServerPropagation
	EC2: DisableSerialConsoleAccess
	EC2: DisableSnapshotBlockPublicAccess
	EC2: DisableTransitGatewayRouteTablePropagation
	EC2: DisableVgwRoutePropagation
	EC2: DisableVpcClassicLink
	EC2: DisableVpcClassicLinkDnsSupport
	EC2: DisassociateAddress
	EC2: DisassociateCapacityReservationBillingOwner
	EC2: DisassociateClientVpnTargetNetwork
	EC2: DisassociateEnclaveCertificateIamRole
	EC2: DisassociateIamInstanceProfile
	EC2: DisassociateInstanceEventWindow
	EC2: DisassociateIpamByoasn
	EC2: DisassociateIpamResourceDiscovery
	EC2: DisassociateNatGatewayAddress
	EC2: DisassociateRouteServer
	EC2: DisassociateRouteTable
	EC2: DisassociateSecurityGroupVpc

Awalan layanan	Tindakan
	EC2: DisassociateSubnetCidrBlock
	EC2: DisassociateTransitGatewayMulticastDomain
	EC2: DisassociateTransitGatewayPolicyTable
	EC2: DisassociateTransitGatewayRouteTable
	EC2: DisassociateTrunkInterface
	EC2: DisassociateVpcCidrBlock
	EC2: EnableAddressTransfer
	EC2: EnableAllowedImagesSettings
	EC2: EnableAwsNetworkPerformanceMetricSubscription
	EC2: EnableCapacityManager
	EC2: EnableEbsEncryptionByDefault
	EC2: EnableFastLaunch
	EC2: EnableFastSnapshotRestores
	EC2: EnableImage
	EC2: EnableImageBlockPublicAccess
	EC2: EnableImageDeprecation
	EC2: EnableImageDeregistrationProtection
	EC2: EnableInstanceSqlHaStandbyDetections
	EC2: EnableIpamOrganizationAdminAccount
	EC2: EnableIpamPolicy
	EC2: EnableReachabilityAnalyzerOrganizationSharing

Awalan layanan	Tindakan
	EC2: EnableRouteServerPropagation
	EC2: EnableSerialConsoleAccess
	EC2: EnableSnapshotBlockPublicAccess
	EC2: EnableTransitGatewayRouteTablePropagation
	EC2: EnableVgwRoutePropagation
	ec2: EnableVolume IO
	EC2: EnableVpcClassicLink
	EC2: EnableVpcClassicLinkDnsSupport
	EC2: ExportClientVpnClientCertificateRevocationList
	EC2: ExportClientVpnClientConfiguration
	EC2: ExportImage
	EC2: ExportTransitGatewayRoutes
	EC2: ExportVerifiedAccessInstanceClientConfiguration
	EC2: GetActiveVpnTunnelStatus
	EC2: GetAllowedImagesSettings
	EC2: GetAssociatedEnclaveCertificateIamRoles
	EC2: GetAssociatedIpv6PoolCidrs
	EC2: GetAwsNetworkPerformanceData
	EC2: GetCapacityManagerAttributes
	EC2: GetCapacityManagerMetricData
	EC2: GetCapacityManagerMetricDimensions

Awalan layanan	Tindakan
	EC2: GetCapacityReservationUsage
	EC2: GetCoipPoolUsage
	EC2: GetConsoleOutput
	EC2: GetConsoleScreenshot
	EC2: GetDeclarativePoliciesReportSummary
	EC2: GetDefaultCreditSpecification
	EC2: GetEbsDefaultKmsKeyId
	EC2: GetEbsEncryptionByDefault
	EC2: GetEnabledIpamPolicy
	EC2: GetFlowLogsIntegrationTemplate
	EC2: GetGroupsForCapacityReservation
	EC2: GetHostReservationPurchasePreview
	EC2: GetImageAncestry
	EC2: GetImageBlockPublicAccessState
	EC2: GetInstanceMetadataDefaults
	EC2: GetInstanceTpmEkPub
	EC2: GetInstanceTypesFromInstanceRequirements
	EC2: GetInstanceUefiData
	EC2: GetIpamAddressHistory
	EC2: GetIpamDiscoveredAccounts
	EC2: GetIpamDiscoveredPublicAddresses

Awalan layanan	Tindakan
	EC2: GetIpamDiscoveredResourceCidrs
	EC2: GetIpamPolicyAllocationRules
	EC2: GetIpamPolicyOrganizationTargets
	EC2: GetIpamPoolAllocations
	EC2: GetIpamPoolCidrs
	EC2: GetIpamPrefixListResolverRules
	EC2: GetIpamPrefixListResolverVersionEntries
	EC2: GetIpamPrefixListResolverVersions
	EC2: GetIpamResourceCidrs
	EC2: GetLaunchTemplateData
	EC2: GetManagedPrefixListAssociations
	EC2: GetManagedPrefixListEntries
	EC2: GetManagedResourceVisibility
	EC2: GetNetworkInsightsAccessScopeAnalysisFindings
	EC2: GetNetworkInsightsAccessScopeContent
	EC2: GetPasswordData
	EC2: GetReservedInstancesExchangeQuote
	EC2: GetRouteServerAssociations
	EC2: GetRouteServerPropagations
	EC2: GetRouteServerRoutingDatabase
	EC2: GetSecurityGroupsForVpc

Awalan layanan	Tindakan
	EC2: GetSerialConsoleAccessStatus
	EC2: GetSnapshotBlockPublicAccessState
	EC2: GetSpotPlacementScores
	EC2: GetSubnetCidrReservations
	EC2: GetTransitGatewayAttachmentPropagations
	EC2: GetTransitGatewayMeteringPolicyEntries
	EC2: GetTransitGatewayMulticastDomainAssociations
	EC2: GetTransitGatewayPolicyTableAssociations
	EC2: GetTransitGatewayPolicyTableEntries
	EC2: GetTransitGatewayPrefixListReferences
	EC2: GetTransitGatewayRouteTableAssociations
	EC2: GetTransitGatewayRouteTablePropagations
	EC2: GetVerifiedAccessEndpointPolicy
	EC2: GetVerifiedAccessEndpointTargets
	EC2: GetVerifiedAccessGroupPolicy
	EC2: GetVpcResourcesBlockingEncryptionEnforcement
	EC2: GetVpnConnectionDeviceSampleConfiguration
	EC2: GetVpnConnectionDeviceTypes
	EC2: GetVpnTunnelReplacementStatus
	EC2: ImportClientVpnClientCertificateRevocationList
	EC2: ImportImage

Awalan layanan	Tindakan
	EC2: ImportInstance
	EC2: ImportKeyPair
	EC2: ImportSnapshot
	EC2: ImportVolume
	EC2: ListImagesInRecycleBin
	EC2: ListSnapshotsInRecycleBin
	EC2: ListVolumesInRecycleBin
	EC2: LockSnapshot
	EC2: ModifyAddressAttribute
	EC2: ModifyAvailabilityZoneGroup
	EC2: ModifyCapacityReservation
	EC2: ModifyCapacityReservationFleet
	EC2: ModifyClientVpnEndpoint
	EC2: ModifyDefaultCreditSpecification
	EC2: ModifyEbsDefaultKmsKeyId
	EC2: ModifyFleet
	EC2: ModifyFpgaImageAttribute
	EC2: ModifyHosts
	EC2: ModifyIdFormat
	EC2: ModifyIdentityIdFormat
	EC2: ModifyImageAttribute

Awalan layanan	Tindakan
	EC2: ModifyInstanceAttribute
	EC2: ModifyInstanceCapacityReservationAttributes
	EC2: ModifyInstanceConnectEndpoint
	EC2: ModifyInstanceCpuOptions
	EC2: ModifyInstanceCreditSpecification
	EC2: ModifyInstanceEventStartTime
	EC2: ModifyInstanceEventWindow
	EC2: ModifyInstanceMaintenanceOptions
	EC2: ModifyInstanceMetadataDefaults
	EC2: ModifyInstanceMetadataOptions
	EC2: ModifyInstanceNetworkPerformanceOptions
	EC2: ModifyInstancePlacement
	EC2: ModifyIpam
	EC2: ModifyIpamPolicyAllocationRules
	EC2: ModifyIpamPool
	EC2: ModifyIpamPoolAllocation
	EC2: ModifyIpamPrefixListResolver
	EC2: ModifyIpamPrefixListResolverTarget
	EC2: ModifyIpamResourceCidr
	EC2: ModifyIpamResourceDiscovery
	EC2: ModifyIpamScope

Awalan layanan	Tindakan
	EC2: ModifyLaunchTemplate
	EC2: ModifyLocalGatewayRoute
	EC2: ModifyManagedPrefixList
	EC2: ModifyManagedResourceVisibility
	EC2: ModifyNetworkInterfaceAttribute
	EC2: ModifyPrivateDnsNameOptions
	EC2: ModifyPublicIpDnsNameOptions
	EC2: ModifyReservedInstances
	EC2: ModifyRouteServer
	EC2: ModifySecurityGroupRules
	EC2: ModifySnapshotAttribute
	EC2: ModifySnapshotTier
	EC2: ModifySpotFleetRequest
	EC2: ModifySubnetAttribute
	EC2: ModifyTrafficMirrorFilterNetworkServices
	EC2: ModifyTrafficMirrorFilterRule
	EC2: ModifyTrafficMirrorSession
	EC2: ModifyTransitGateway
	EC2: ModifyTransitGatewayMeteringPolicy
	EC2: ModifyTransitGatewayPrefixListReference
	EC2: ModifyTransitGatewayVpcAttachment

Awalan layanan	Tindakan
	EC2: ModifyVerifiedAccessEndpoint
	EC2: ModifyVerifiedAccessEndpointPolicy
	EC2: ModifyVerifiedAccessGroup
	EC2: ModifyVerifiedAccessGroupPolicy
	EC2: ModifyVerifiedAccessInstance
	EC2: ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2: ModifyVerifiedAccessTrustProvider
	EC2: ModifyVolume
	EC2: ModifyVolumeAttribute
	EC2: ModifyVpcAttribute
	EC2: ModifyVpcBlockPublicAccessExclusion
	EC2: ModifyVpcBlockPublicAccessOptions
	EC2: ModifyVpcEncryptionControl
	EC2: ModifyVpcEndpoint
	EC2: ModifyVpcEndpointConnectionNotification
	EC2: ModifyVpcEndpointServiceConfiguration
	EC2: ModifyVpcEndpointServicePayerResponsibility
	EC2: ModifyVpcEndpointServicePermissions
	EC2: ModifyVpcPeeringConnectionOptions
	EC2: ModifyVpcTenancy
	EC2: ModifyVpnConnection

Awalan layanan	Tindakan
	EC2: ModifyVpnConnectionOptions
	EC2: ModifyVpnTunnelCertificate
	EC2: ModifyVpnTunnelOptions
	EC2: MonitorInstances
	EC2: MoveAddressToVpc
	EC2: MoveByoipCidrToIpam
	EC2: MoveCapacityReservationInstances
	EC2: ProvisionByoipCidr
	EC2: ProvisionIpamByoasn
	EC2: ProvisionIpamPoolCidr
	EC2: ProvisionPublicIpv4PoolCidr
	EC2: PurchaseCapacityBlockExtension
	EC2: PurchaseHostReservation
	EC2: PurchaseReservedInstancesOffering
	EC2: PurchaseScheduledInstances
	EC2: RebootInstances
	EC2: RegisterImage
	EC2: RegisterInstanceEventNotificationAttributes
	EC2: RegisterTransitGatewayMulticastGroupMembers
	EC2: RegisterTransitGatewayMulticastGroupSources
	EC2: RejectCapacityReservationBillingOwnership

Awalan layanan	Tindakan
	EC2: RejectTransitGatewayClientVpnAttachment
	EC2: RejectTransitGatewayMulticastDomainAssociations
	EC2: RejectTransitGatewayPeeringAttachment
	EC2: RejectTransitGatewayVpcAttachment
	EC2: RejectVpcEndpointConnections
	EC2: RejectVpcPeeringConnection
	EC2: ReleaseAddress
	EC2: ReleaseHosts
	EC2: ReleaseIpamPoolAllocation
	EC2: ReplaceIamInstanceProfileAssociation
	EC2: ReplaceImageCriteriaInAllowedImagesSettings
	EC2: ReplaceNetworkAclAssociation
	EC2: ReplaceNetworkAclEntry
	EC2: ReplaceRoute
	EC2: ReplaceRouteTableAssociation
	EC2: ReplaceTransitGatewayRoute
	EC2: ReplaceVpnTunnel
	EC2: ReportInstanceStatus
	EC2: RequestSpotFleet
	EC2: RequestSpotInstances
	EC2: ResetAddressAttribute

Awalan layanan	Tindakan
	EC2: ResetEbsDefaultKmsKeyId
	EC2: ResetFpgaImageAttribute
	EC2: ResetImageAttribute
	EC2: ResetInstanceAttribute
	EC2: ResetNetworkInterfaceAttribute
	EC2: ResetSnapshotAttribute
	EC2: RestoreAddressToClassic
	EC2: RestoreImageFromRecycleBin
	EC2: RestoreManagedPrefixListVersion
	EC2: RestoreSnapshotFromRecycleBin
	EC2: RestoreSnapshotTier
	EC2: RestoreVolumeFromRecycleBin
	EC2: RevokeClientVpnIngress
	EC2: RevokeSecurityGroupEgress
	EC2: RevokeSecurityGroupIngress
	EC2: RunInstances
	EC2: RunScheduledInstances
	EC2: SearchLocalGatewayRoutes
	EC2: SearchTransitGatewayMulticastGroups
	EC2: SearchTransitGatewayRoutes
	EC2: SendDiagnosticInterrupt

Awalan layanan	Tindakan
	EC2: StartDeclarativePoliciesReport
	EC2: StartInstances
	EC2: StartNetworkInsightsAccessScopeAnalysis
	EC2: StartNetworkInsightsAnalysis
	EC2: StartVpcEndpointServicePrivateDnsVerification
	EC2: TerminateClientVpnConnections
	EC2: UnassignIpv6Addresses
	EC2: UnassignPrivateIpAddresses
	EC2: UnassignPrivateNatGatewayAddress
	EC2: UnlockSnapshot
	EC2: UnmonitorInstances
	EC2: UpdateCapacityManagerOrganizationsAccess
	EC2: UpdateInterruptibleCapacityReservationAllocation
	EC2: UpdateSecurityGroupRuleDescriptionsEgress
	EC2: UpdateSecurityGroupRuleDescriptionsIngress
	EC2: WithdrawByoipCidr

Awalan layanan	Tindakan
ecr	ecr: BatchCheckLayerAvailability ecr: BatchDeleteImage ecr: BatchGetImage ecr: BatchGetRepositoryScanningConfiguration ecr: CompleteLayerUpload ecr: CreatePullThroughCacheRule ecr: CreateRepositoryCreationTemplate ecr: DeleteLifecyclePolicy ecr: DeletePullThroughCacheRule ecr: DeleteRegistryPolicy ecr: DeleteRepository ecr: DeleteRepositoryCreationTemplate ecr: DeleteRepositoryPolicy ecr: DeleteSigningConfiguration ecr: DescribeImageReplicationStatus ecr: DescribeImageScanFindings ecr: DescribeImages ecr: DescribePullThroughCacheRules ecr: DescribeRegistry ecr: DescribeRepositories ecr: DescribeRepositoryCreationTemplates

Awalan layanan	Tindakan
	ecr: GetAccountSetting ecr: GetAuthorizationToken ecr: GetDownloadUrlForLayer ecr: GetLifecyclePolicy ecr: GetLifecyclePolicyPreview ecr: GetRegistryPolicy ecr: GetRegistryScanningConfiguration ecr: GetRepositoryPolicy ecr: GetSigningConfiguration ecr: InitiateLayerUpload ecr: ListImages ecr: ListPullTimeUpdateExclusions ecr: PutAccountSetting ecr: PutImage ecr: PutImageScanningConfiguration ecr: PutRegistryPolicy ecr: PutRegistryScanningConfiguration ecr: PutReplicationConfiguration ecr: StartImageScan ecr: StartLifecyclePolicyPreview ecr: UpdatePullThroughCacheRule

Awalan layanan	Tindakan
	ecr: UpdateRepositoryCreationTemplate ecr: UploadLayerPart ecr: ValidatePullThroughCacheRule

Awalan layanan	Tindakan
ecr-publik	ecr-publik: BatchCheckLayerAvailability
	ecr-publik: BatchDeleteImage
	ecr-publik: CompleteLayerUpload
	ecr-publik: CreateRepository
	ecr-publik: DeleteRepository
	ecr-publik: DeleteRepositoryPolicy
	ecr-publik: DescribeImages
	ecr-publik: DescribeRegistries
	ecr-publik: DescribeRepositories
	ecr-publik: GetAuthorizationToken
	ecr-publik: GetRegistryCatalogData
	ecr-publik: GetRepositoryCatalogData
	ecr-publik: GetRepositoryPolicy
	ecr-publik: InitiateLayerUpload
	ecr-publik: PutImage
	ecr-publik: PutRegistryCatalogData
	ecr-publik: PutRepositoryCatalogData
	ecr-publik: SetRepositoryPolicy
	ecr-publik: UploadLayerPart

Awalan layanan	Tindakan
ecs	ecs: ContinueServiceDeployment ecs: CreateCapacityProvider ecs: CreateCluster ecs: CreateDaemon ecs: CreateService ecs: CreateTaskSet ecs: DeleteAccountSetting ecs: DeleteAttributes ecs: DeleteCapacityProvider ecs: DeleteCluster ecs: DeleteDaemon ecs: DeleteDaemonTaskDefinition ecs: DeleteExpressGatewayService ecs: DeleteService ecs: DeleteTaskDefinitions ecs: DeleteTaskSet ecs: DeregisterContainerInstance ecs: DeregisterTaskDefinition ecs: DescribeCapacityProviders ecs: DescribeClusters ecs: DescribeContainerInstances

Awalan layanan	Tindakan
	ecs: DescribeDaemon ecs: DescribeDaemonDeployments ecs: DescribeDaemonRevisions ecs: DescribeDaemonTaskDefinition ecs: DescribeExpressGatewayService ecs: DescribeServiceDeployments ecs: DescribeServiceRevisions ecs: DescribeServices ecs: DescribeTaskDefinition ecs: DescribeTaskSets ecs: DescribeTasks ecs: DiscoverPollEndpoint ecs: ExecuteCommand ecs: GetTaskProtection ecs: ListAccountSettings ecs: ListAttributes ecs: ListClusters ecs: ListContainerInstances ecs: ListDaemonDeployments ecs: ListDaemonTaskDefinitions ecs: ListDaemons

Awalan layanan	Tindakan
	ecs: ListServiceDeployments ecs: ListServices ecs: ListServicesByNamespace ecs: ListTaskDefinitionFamilies ecs: ListTaskDefinitions ecs: ListTasks ecs: PutAccountSetting ecs: PutAccountSettingDefault ecs: PutAttributes ecs: PutClusterCapacityProviders ecs: RegisterContainerInstance ecs: RegisterDaemonTaskDefinition ecs: RunTask ecs: StartTask ecs: StopServiceDeployment ecs: StopTask ecs: SubmitAttachmentStateChanges ecs: SubmitContainerStateChange ecs: SubmitTaskStateChange ecs: UpdateCapacityProvider ecs: UpdateCluster

Awalan layanan	Tindakan
	ecs: UpdateClusterSettings ecs: UpdateContainerAgent ecs: UpdateContainerInstancesState ecs: UpdateDaemon ecs: UpdateExpressGatewayService ecs: UpdateService ecs: UpdateServicePrimaryTaskSet ecs: UpdateTaskProtection ecs: UpdateTaskSet

Awalan layanan	Tindakan
eks	eks: AssociateAccessPolicy
	eks: AssociateEncryptionConfig
	eks: AssociateIdentityProviderConfig
	eks: CancelUpdate
	eks: CreateAccessEntry
	eks: CreateAddon
	eks: CreateCluster
	eks: CreateEksAnywhereSubscription
	eks: CreateFargateProfile
	eks: CreateNodegroup
	eks: DeleteAccessEntry
	eks: DeleteAddon
	eks: DeleteCapability
	eks: DeleteCluster
	eks: DeleteEksAnywhereSubscription
	eks: DeleteFargateProfile
	eks: DeleteNodegroup
	eks: DeletePodIdentityAssociation
	eks: DeregisterCluster
	eks: DescribeAccessEntry
	eks: DescribeAddon

Awalan layanan	Tindakan
	eks: DescribeAddonConfiguration
	eks: DescribeAddonVersions
	eks: DescribeCapability
	eks: DescribeCluster
	eks: DescribeClusterVersions
	eks: DescribeEksAnywhereSubscription
	eks: DescribeFargateProfile
	eks: DescribeIdentityProviderConfig
	eks: DescribeInsight
	eks: DescribeInsightsRefresh
	eks: DescribeNodegroup
	eks: DescribePodIdentityAssociation
	eks: DescribeUpdate
	eks: DisassociateAccessPolicy
	eks: DisassociateIdentityProviderConfig
	eks: ListAccessEntries
	eks: ListAccessPolicies
	eks: ListAddons
	eks: ListAssociatedAccessPolicies
	eks: ListCapabilities
	eks: ListClusters

Awalan layanan	Tindakan
	eks: ListEksAnywhereSubscriptions
	eks: ListFargateProfiles
	eks: ListIdentityProviderConfigs
	eks: ListInsights
	eks: ListNodegroups
	eks: ListPodIdentityAssociations
	eks: ListUpdates
	eks: RegisterCluster
	eks: StartInsightsRefresh
	eks: UpdateAccessEntry
	eks: UpdateAddon
	eks: UpdateCapability
	eks: UpdateClusterConfig
	eks: UpdateClusterVersion
	eks: UpdateEksAnywhereSubscription
	eks: UpdateNodegroupConfig
	eks: UpdateNodegroupVersion
	eks: UpdatePodIdentityAssociation

Awalan layanan	Tindakan
elasticache	elastisakit: AuthorizeCacheSecurityGroupIngress elastisakit: BatchApplyUpdateAction elastisakit: BatchStopUpdateAction elastisakit: CompleteMigration elastisakit: CopyServerlessCacheSnapshot elastisakit: CopySnapshot elastisakit: CreateCacheCluster elastisakit: CreateCacheParameterGroup elastisakit: CreateCacheSecurityGroup elastisakit: CreateCacheSubnetGroup elastisakit: CreateGlobalReplicationGroup elastisakit: CreateReplicationGroup elastisakit: CreateServerlessCache elastisakit: CreateServerlessCacheSnapshot elastisakit: CreateSnapshot elastisakit: CreateUser elastisakit: CreateUserGroup elastisakit: DecreaseNodeGroupsInGlobalReplicationGroup elastisakit: DecreaseReplicaCount elastisakit: DeleteCacheCluster elastisakit: DeleteCacheParameterGroup

Awalan layanan	Tindakan
	elastisakit: DeleteCacheSecurityGroup elastisakit: DeleteCacheSubnetGroup elastisakit: DeleteGlobalReplicationGroup elastisakit: DeleteReplicationGroup elastisakit: DeleteServerlessCache elastisakit: DeleteServerlessCacheSnapshot elastisakit: DeleteSnapshot elastisakit: DeleteUser elastisakit: DeleteUserGroup elastisakit: DescribeCacheClusters elastisakit: DescribeCacheEngineVersions elastisakit: DescribeCacheParameterGroups elastisakit: DescribeCacheParameters elastisakit: DescribeCacheSecurityGroups elastisakit: DescribeCacheSubnetGroups elastisakit: DescribeEngineDefaultParameters elastisakit: DescribeEvents elastisakit: DescribeGlobalReplicationGroups elastisakit: DescribeReplicationGroups elastisakit: DescribeReservedCacheNodes elastisakit: DescribeReservedCacheNodesOfferings

Awalan layanan	Tindakan
	elastisakit: DescribeServerlessCacheSnapshots elastisakit: DescribeServerlessCaches elastisakit: DescribeServiceUpdates elastisakit: DescribeSnapshots elastisakit: DescribeUpdateActions elastisakit: DescribeUserGroups elastisakit: DescribeUsers elastisakit: DisassociateGlobalReplicationGroup elastisakit: ExportServerlessCacheSnapshot elastisakit: FailoverGlobalReplicationGroup elastisakit: IncreaseNodeGroupsInGlobalReplicationGroup elastisakit: IncreaseReplicaCount elastisakit: ListAllowedNodeTypeModifications elastisakit: ModifyCacheCluster elastisakit: ModifyCacheParameterGroup elastisakit: ModifyCacheSubnetGroup elastisakit: ModifyGlobalReplicationGroup elastisakit: ModifyReplicationGroup elastisakit: ModifyReplicationGroupShardConfiguration elastisakit: ModifyServerlessCache elastisakit: ModifyUser

Awalan layanan	Tindakan
	elastisakit: ModifyUserGroup elastisakit: PurchaseReservedCacheNodesOffering elastisakit: RebalanceSlotsInGlobalReplicationGroup elastisakit: RebootCacheCluster elastisakit: ResetCacheParameterGroup elastisakit: RevokeCacheSecurityGroupIngress elastisakit: StartMigration elastisakit: TestFailover elastisakit: TestMigration

Awalan layanan	Tindakan
tangkai beanstalk	tangkai elastis: AbortEnvironmentUpdate
	tangkai elastis: ApplyEnvironmentManagedAction
	tangkai elastis: AssociateEnvironmentOperationsRole
	ElasticBeanstalk:CheckDNSAvailability
	tangkai elastis: ComposeEnvironments
	tangkai elastis: CreateApplication
	tangkai elastis: CreateApplicationVersion
	tangkai elastis: CreateConfigurationTemplate
	tangkai elastis: CreateEnvironment
	tangkai elastis: CreatePlatformVersion
	tangkai elastis: CreateStorageLocation
	tangkai elastis: DeleteApplication
	tangkai elastis: DeleteApplicationVersion
	tangkai elastis: DeleteConfigurationTemplate
	tangkai elastis: DeleteEnvironmentConfiguration
	tangkai elastis: DeletePlatformVersion
	tangkai elastis: DescribeAccountAttributes
	tangkai elastis: DescribeApplicationVersions
	tangkai elastis: DescribeApplications
	tangkai elastis: DescribeConfigurationOptions
	tangkai elastis: DescribeConfigurationSettings

Awalan layanan	Tindakan
	tangkai elastis: DescribeEnvironmentHealth
	tangkai elastis: DescribeEnvironmentManagedActionHistory
	tangkai elastis: DescribeEnvironmentManagedActions
	tangkai elastis: DescribeEnvironmentResources
	tangkai elastis: DescribeEnvironments
	tangkai elastis: DescribeEvents
	tangkai elastis: DescribeInstancesHealth
	tangkai elastis: DescribePlatformVersion
	tangkai elastis: DisassociateEnvironmentOperationsRole
	tangkai elastis: ListAvailableSolutionStacks
	tangkai elastis: ListPlatformBranches
	tangkai elastis: ListPlatformVersions
	tangkai elastis: RebuildEnvironment
	tangkai elastis: RequestEnvironmentInfo
	tangkai elastis: RestartAppServer
	tangkai elastis: RetrieveEnvironmentInfo
	tangkai elastis: SwapEnvironmentCNAMEs
	tangkai elastis: TerminateEnvironment
	tangkai elastis: UpdateApplication
	tangkai elastis: UpdateApplicationResourceLifecycle
	tangkai elastis: UpdateApplicationVersion

Awalan layanan	Tindakan
	tangkai elastis: UpdateConfigurationTemplate tangkai elastis: UpdateEnvironment tangkai elastis: ValidateConfigurationSettings

Awalan layanan	Tindakan
sistem file elastis	sistem file elastis: CreateAccessPoint sistem file elastis: CreateFileSystem sistem file elastis: CreateMountTarget sistem file elastis: CreateReplicationConfiguration sistem file elastis: DeleteAccessPoint sistem file elastis: DeleteFileSystem sistem file elastis: DeleteFileSystemPolicy sistem file elastis: DeleteMountTarget sistem file elastis: DeleteReplicationConfiguration sistem file elastis: DescribeAccessPoints sistem file elastis: DescribeAccountPreferences sistem file elastis: DescribeBackupPolicy sistem file elastis: DescribeFileSystemPolicy sistem file elastis: DescribeFileSystems sistem file elastis: DescribeLifecycleConfiguration sistem file elastis: DescribeMountTargetSecurityGroups sistem file elastis: DescribeMountTargets sistem file elastis: DescribeReplicationConfigurations sistem file elastis: ModifyMountTargetSecurityGroups sistem file elastis: PutAccountPreferences sistem file elastis: PutBackupPolicy

Awalan layanan	Tindakan
	sistem file elastis: PutFileSystemPolicy sistem file elastis: PutLifecycleConfiguration sistem file elastis: UpdateFileSystem sistem file elastis: UpdateFileSystemProtection

Awalan layanan	Tindakan
elasticloadbalancing	elasticloadbalancing: AddListenerCertificates
	elasticloadbalancing: AddTrustStoreRevocations
	elasticloadbalancing: ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing: AttachLoadBalancerToSubnets
	elasticloadbalancing: ConfigureHealthCheck
	elasticloadbalancing: CreateAppCookieStickinessPolicy
	elasticloadbalancing: CreateLBCookieStickinessPolicy
	elasticloadbalancing: CreateListener
	elasticloadbalancing: CreateLoadBalancer
	elasticloadbalancing: CreateLoadBalancerListeners
	elasticloadbalancing: CreateLoadBalancerPolicy
	elasticloadbalancing: CreateRule
	elasticloadbalancing: CreateTargetGroup
	elasticloadbalancing: CreateTrustStore
	elasticloadbalancing: DeleteListener
	elasticloadbalancing: DeleteLoadBalancer
	elasticloadbalancing: DeleteLoadBalancerListeners
	elasticloadbalancing: DeleteLoadBalancerPolicy
	elasticloadbalancing: DeleteRule
	elasticloadbalancing: DeleteSharedTrustStoreAssociation
	elasticloadbalancing: DeleteTargetGroup

Awalan layanan	Tindakan
	elasticloadbalancing: DeleteTrustStore
	elasticloadbalancing: DeregisterInstancesFromLoadBalancer
	elasticloadbalancing: DeregisterTargets
	elasticloadbalancing: DescribeAccountLimits
	elasticloadbalancing: DescribeCapacityReservation
	elasticloadbalancing: DescribeInstanceHealth
	elasticloadbalancing: DescribeListenerAttributes
	elasticloadbalancing: DescribeListenerCertificates
	elasticloadbalancing: DescribeListeners
	elasticloadbalancing: DescribeLoadBalancerAttributes
	elasticloadbalancing: DescribeLoadBalancerPolicies
	elasticloadbalancing: DescribeLoadBalancerPolicyTypes
	elasticloadbalancing: DescribeLoadBalancers
	elasticloadbalancing: DescribeRules
	ElasticLoadBalancing:DescribessIPolicies
	elasticloadbalancing: DescribeTargetGroupAttributes
	elasticloadbalancing: DescribeTargetGroups
	elasticloadbalancing: DescribeTargetHealth
	elasticloadbalancing: DescribeTrustStoreAssociations
	elasticloadbalancing: DescribeTrustStoreRevocations
	elasticloadbalancing: DescribeTrustStores

Awalan layanan	Tindakan
	elasticloadbalancing: DetachLoadBalancerFromSubnets
	elasticloadbalancing: DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: GetResourcePolicy
	elasticloadbalancing: GetTrustStoreCaCertificatesBundle
	elasticloadbalancing: GetTrustStoreRevocationContent
	elasticloadbalancing: ModifyCapacityReservation
	elasticloadbalancing: ModifyIpPools
	elasticloadbalancing: ModifyListener
	elasticloadbalancing: ModifyLoadBalancerAttributes
	elasticloadbalancing: ModifyRule
	elasticloadbalancing: ModifyTargetGroup
	elasticloadbalancing: ModifyTargetGroupAttributes
	elasticloadbalancing: ModifyTrustStore
	elasticloadbalancing: RegisterInstancesWithLoadBalancer
	elasticloadbalancing: RegisterTargets
	elasticloadbalancing: RemoveListenerCertificates
	elasticloadbalancing: RemoveTrustStoreRevocations
	elasticloadbalancing: SetIpAddressType
	elasticloadbalancing: SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing: SetLoadBalancerPoliciesForBackendServer

Awalan layanan	Tindakan
	elasticloadbalancing: SetLoadBalancerPoliciesOfListener
	elasticloadbalancing: SetRulePriorities
	elasticloadbalancing: SetSecurityGroups
	elasticloadbalancing: SetSubnets

Awalan layanan	Tindakan
wadah emr	wadah emr: CancelJobRun wadah emr: CreateJobTemplate wadah emr: CreateManagedEndpoint wadah emr: CreateSecurityConfiguration wadah emr: CreateVirtualCluster wadah emr: DeleteJobTemplate wadah emr: DeleteManagedEndpoint wadah emr: DeleteSecurityConfiguration wadah emr: DeleteVirtualCluster wadah emr: DescribeJobRun emr-kontainer: DescribeJobTemplate emr-kontainer: DescribeManagedEndpoint emr-kontainer: DescribeSecurityConfiguration emr-kontainer: DescribeVirtualCluster emr-kontainer: GetManagedEndpointSessionCredentials emr-kontainer: ListJobRuns emr-kontainer: ListJobTemplates emr-kontainer: ListManagedEndpoints emr-kontainer: ListSecurityConfigurations emr-kontainer: ListVirtualClusters emr-kontainer: StartJobRun

Awalan layanan	Tindakan
emr-tanpa server	emr-tanpa server: CancelJobRun
	emr-tanpa server: CreateApplication
	emr-tanpa server: DeleteApplication
	emr-tanpa server: GetApplication
	emr-tanpa server: GetDashboardForJobRun
	emr-tanpa server: GetJobRun
	emr-tanpa server: GetResourceDashboard
	emr-tanpa server: GetSession
	emr-tanpa server: GetSessionEndpoint
	emr-tanpa server: ListApplications
	emr-tanpa server: ListJobRunAttempts
	emr-tanpa server: ListJobRuns
	emr-tanpa server: ListSessions
	emr-tanpa server: StartApplication
	emr-tanpa server: StartJobRun
	emr-tanpa server: StopApplication
	emr-tanpa server: TerminateSession
	emr-tanpa server: UpdateApplication

Awalan layanan	Tindakan
es	es: AcceptInboundConnection es: AcceptInboundCrossClusterSearchConnection es: AssociatePackage es: AuthorizeVpcEndpointAccess es: CancelElasticsearchServiceSoftwareUpdate es: CancelServiceSoftwareUpdate es: CreateDomain es: CreateElasticsearchDomain es: CreateIndex es: CreateOutboundConnection es: CreateOutboundCrossClusterSearchConnection es: CreatePackage es: CreateVpcEndpoint es: DeleteDomain es: DeleteElasticsearchDomain es: DeleteElasticsearchServiceRole es: DeleteInboundConnection es: DeleteInboundCrossClusterSearchConnection es: DeleteIndex es: DeleteOutboundConnection es: DeleteOutboundCrossClusterSearchConnection

Awalan layanan	Tindakan
	es: DeletePackage es: DeleteVpcEndpoint es: DescribeDomain es: DescribeDomainAutoTunes es: DescribeDomainChangeProgress es: DescribeDomainConfig es: DescribeDomainHealth es: DescribeDomainNodes es: DescribeDomains es: DescribeDryRunProgress es: DescribeElasticsearchDomain es: DescribeElasticsearchDomainConfig es: DescribeElasticsearchDomains es: DescribeElasticsearchInstanceTypeLimits es: DescribeInboundConnections es: DescribeInboundCrossClusterSearchConnections es: DescribeInstanceTypeLimits es: DescribeOutboundConnections es: DescribeOutboundCrossClusterSearchConnections es: DescribePackages es: DescribeReservedElasticsearchInstanceOfferings

Awalan layanan	Tindakan
	es: DescribeReservedElasticsearchInstances es: DescribeReservedInstanceOfferings es: DescribeReservedInstances es: DescribeVpcEndpoints es: DissociatePackage es: DissociatePackages es: GetCompatibleElasticsearchVersions es: GetCompatibleVersions es: GetDataSource es: GetDomainMaintenanceStatus es: GetPackageVersionHistory es: GetUpgradeHistory es: GetUpgradeStatus es: ListDataSources es: ListDomainNames es: ListDomainsForPackage es: ListElasticsearchInstanceTypes es: ListElasticsearchVersions es: ListInsights es: ListInstanceTypeDetails es: ListPackagesForDomain

Awalan layanan	Tindakan
	es: ListScheduledActions es: ListVersions es: ListVpcEndpointAccess es: ListVpcEndpoints es: ListVpcEndpointsForDomain es: PurchaseReservedElasticsearchInstanceOffering es: PurchaseReservedInstanceOffering es: RejectInboundConnection es: RejectInboundCrossClusterSearchConnection es: RevokeVpcEndpointAccess es: RollbackServiceSoftwareUpdate es: StartDomainMaintenance es: StartElasticsearchServiceSoftwareUpdate es: StartServiceSoftwareUpdate es: UpdateDataSource es: UpdateDomainConfig es: UpdateElasticsearchDomainConfig es: UpdateIndex es: UpdatePackage es: UpdatePackageScope es: UpdateScheduledAction

Awalan layanan	Tindakan
	es: UpdateVpcEndpoint es: UpgradeDomain es: UpgradeElasticsearchDomain

Awalan layanan	Tindakan
peristiwa	peristiwa: ActivateEventSource
	peristiwa: CancelReplay
	peristiwa: CreateApiDestination
	peristiwa: CreateArchive
	peristiwa: CreateConnection
	peristiwa: CreateEndpoint
	peristiwa: CreateEventBus
	peristiwa: CreatePartnerEventSource
	peristiwa: DeactivateEventSource
	peristiwa: DeauthorizeConnection
	peristiwa: DeleteApiDestination
	peristiwa: DeleteArchive
	peristiwa: DeleteConnection
	peristiwa: DeleteEndpoint
	peristiwa: DeleteEventBus
	peristiwa: DeletePartnerEventSource
	peristiwa: DeleteRule
	peristiwa: DescribeApiDestination
	peristiwa: DescribeArchive
	peristiwa: DescribeConnection
	peristiwa: DescribeEndpoint

Awalan layanan	Tindakan
	peristiwa: DescribeEventBus
	peristiwa: DescribeEventSource
	peristiwa: DescribePartnerEventSource
	peristiwa: DescribeReplay
	peristiwa: DescribeRule
	peristiwa: DisableRule
	peristiwa: EnableRule
	peristiwa: ListApiDestinations
	peristiwa: ListArchives
	peristiwa: ListConnections
	peristiwa: ListEndpoints
	peristiwa: ListEventBuses
	peristiwa: ListEventSources
	peristiwa: ListPartnerEventSourceAccounts
	peristiwa: ListPartnerEventSources
	peristiwa: ListReplays
	peristiwa: ListRuleNamesByTarget
	peristiwa: ListRules
	peristiwa: ListTargetsByRule
	peristiwa: PutPermission
	peristiwa: PutRule

Awalan layanan	Tindakan
	peristiwa: PutTargets peristiwa: RemovePermission peristiwa: RemoveTargets peristiwa: StartReplay peristiwa: TestEventPattern peristiwa: UpdateApiDestination peristiwa: UpdateArchive peristiwa: UpdateConnection peristiwa: UpdateEndpoint peristiwa: UpdateEventBus

Awalan layanan	Tindakan
ruang finspace	ruang finspace: CreateEnvironment ruang finspace: CreateKxChangeset ruang finspace: CreateKxCluster ruang finspace: CreateKxDatabase ruang finspace: CreateKxDataview ruang finspace: CreateKxEnvironment ruang finspace: CreateKxScalingGroup ruang finspace: CreateKxUser ruang finspace: CreateKxVolume ruang finspace: CreateUser ruang finspace: DeleteEnvironment ruang finspace: DeleteKxCluster ruang finspace: DeleteKxClusterNode ruang finspace: DeleteKxDatabase ruang finspace: DeleteKxDataview ruang finspace: DeleteKxEnvironment ruang finspace: DeleteKxScalingGroup ruang finspace: DeleteKxUser ruang finspace: DeleteKxVolume ruang finspace: GetEnvironment ruang finspace: GetKxChangeset

Awalan layanan	Tindakan
	ruang finspace: GetKxCluster ruang finspace: GetKxConnectionString ruang finspace: GetKxDatabase ruang finspace: GetKxDataview ruang finspace: GetKxEnvironment ruang finspace: GetKxScalingGroup ruang finspace: GetKxUser ruang finspace: GetKxVolume ruang finspace: GetLoadSampleDataSetGroupIntoEnvironmentStatus ruang finspace: GetUser ruang finspace: ListEnvironments ruang finspace: ListKxChangesets ruang finspace: ListKxClusterNodes ruang finspace: ListKxClusters ruang finspace: ListKxDatabases ruang finspace: ListKxDataviews ruang finspace: ListKxEnvironments ruang finspace: ListKxScalingGroups ruang finspace: ListKxUsers ruang finspace: ListKxVolumes

Awalan layanan	Tindakan
	ruang finspace: ListUsers ruang finspace: LoadSampleDataSetGroupIntoEnvironment ruang finspace: ResetUserPassword ruang finspace: UpdateEnvironment ruang finspace: UpdateKxClusterCodeConfiguration ruang finspace: UpdateKxClusterDatabases ruang finspace: UpdateKxDatabase ruang finspace: UpdateKxDataview ruang finspace: UpdateKxEnvironment ruang finspace: UpdateKxEnvironmentNetwork ruang finspace: UpdateKxUser ruang finspace: UpdateKxVolume ruang finspace: UpdateUser
firehose	selang api: CreateDeliveryStream selang api: DeleteDeliveryStream selang api: DescribeDeliveryStream selang api: ListDeliveryStreams selang api: StartDeliveryStreamEncryption selang api: StopDeliveryStreamEncryption selang api: UpdateDestination

Awalan layanan	Tindakan
fis	fis: CreateExperimentTemplate fis: CreateTargetAccountConfiguration fis: DeleteExperimentTemplate fis: DeleteTargetAccountConfiguration fis: GetAction fis: GetExperiment fis: GetExperimentTargetAccountConfiguration fis: GetExperimentTemplate fis: GetSafetyLever fis: GetTargetAccountConfiguration fis: GetTargetResourceType fis: ListActions fis: ListExperimentResolvedTargets fis: ListExperimentTargetAccountConfigurations fis: ListExperimentTemplates fis: ListExperiments fis: ListTargetAccountConfigurations fis: ListTargetResourceTypes fis: StartExperiment fis: StopExperiment fis: UpdateExperimentTemplate

Awalan layanan	Tindakan
	fis: UpdateSafetyLeverState fis: UpdateTargetAccountConfiguration

Awalan layanan	Tindakan
fms	fms: AssociateAdminAccount
	fms: AssociateThirdPartyFirewall
	fms: BatchAssociateResource
	fms: BatchDisassociateResource
	fms: DeleteAppsList
	fms: DeleteNotificationChannel
	fms: DeletePolicy
	fms: DeleteProtocolsList
	fms: DeleteResourceSet
	fms: DisassociateAdminAccount
	fms: DisassociateThirdPartyFirewall
	fms: GetAdminAccount
	fms: GetAdminScope
	fms: GetAppsList
	fms: GetComplianceDetail
	fms: GetNotificationChannel
	fms: GetPolicy
	fms: GetProtectionStatus
	fms: GetProtocolsList
	fms: GetResourceSet
	fms: GetThirdPartyFirewallAssociationStatus

Awalan layanan	Tindakan
	fms: GetViolationDetails
	fms: ListAdminAccountsForOrganization
	fms: ListAdminsManagingAccount
	fms: ListAppsLists
	fms: ListComplianceStatus
	fms: ListDiscoveredResources
	fms: ListMemberAccounts
	fms: ListPolicies
	fms: ListProtocolsLists
	fms: ListResourceSetResources
	fms: ListResourceSets
	fms: ListThirdPartyFirewallFirewallPolicies
	fms: PutAdminAccount
	fms: PutAppsList
	fms: PutNotificationChannel
	fms: PutPolicy
	fms: PutProtocolsList
	fms: PutResourceSet

Awalan layanan	Tindakan
detektor penipuan	detektor penipuan: BatchCreateVariable detektor penipuan: BatchGetVariable detektor penipuan: CancelBatchImportJob detektor penipuan: CancelBatchPredictionJob detektor penipuan: CreateBatchImportJob detektor penipuan: CreateBatchPredictionJob detektor penipuan: CreateDetectorVersion detektor penipuan: CreateList detektor penipuan: CreateModel detektor penipuan: CreateModelVersion detektor penipuan: CreateRule detektor penipuan: CreateVariable detektor penipuan: DeleteBatchImportJob detektor penipuan: DeleteBatchPredictionJob detektor penipuan: DeleteDetector detektor penipuan: DeleteDetectorVersion detektor penipuan: DeleteEntityType detektor penipuan: DeleteEvent detektor penipuan: DeleteEventType detektor penipuan: DeleteEventsByEventType detektor penipuan: DeleteExternalModel

Awalan layanan	Tindakan
	detektor penipuan: DeleteLabel detektor penipuan: DeleteList detektor penipuan: DeleteModel detektor penipuan: DeleteModelVersion detektor penipuan: DeleteOutcome detektor penipuan: DeleteRule detektor penipuan: DeleteVariable detektor penipuan: DescribeDetector detektor penipuan: DescribeModelVersions detektor penipuan: GetBatchImportJobs detektor penipuan: GetBatchPredictionJobs detektor penipuan: GetDeleteEventsByEventTypeStatus detektor penipuan: GetDetectorVersion detektor penipuan: GetDetectors detektor penipuan: GetEntityTypes detektor penipuan: GetEvent detektor penipuan: GetEventPrediction detektor penipuan: GetEventPredictionMetadata detektor penipuan: GetEventTypes detektor penipuan: GetExternalModels detektor penipuan: GetKMSEncryptionKey

Awalan layanan	Tindakan
	detektor penipuan: GetLabels detektor penipuan: GetListElements detektor penipuan: GetListsMetadata detektor penipuan: GetModelVersion detektor penipuan: GetModels detektor penipuan: GetOutcomes detektor penipuan: GetRules detektor penipuan: GetVariables detektor penipuan: ListEventPredictions detektor penipuan: PutDetector detektor penipuan: PutEntityType detektor penipuan: PutEventType detektor penipuan: PutExternalModel detektor penipuan: PutKMSEncryptionKey detektor penipuan: PutLabel detektor penipuan: PutOutcome detektor penipuan: SendEvent detektor penipuan: UpdateDetectorVersion detektor penipuan: UpdateDetectorVersionMetadata detektor penipuan: UpdateDetectorVersionStatus detektor penipuan: UpdateEventLabel

Awalan layanan	Tindakan
	detektor penipuan: UpdateList detektor penipuan: UpdateModel detektor penipuan: UpdateModelVersion detektor penipuan: UpdateModelVersionStatus detektor penipuan: UpdateRuleMetadata detektor penipuan: UpdateRuleVersion detektor penipuan: UpdateVariable

Awalan layanan	Tindakan
fsx	fsx: AssociateFileSystemAliases
	fsx: CancelDataRepositoryTask
	fsx: CopyBackup
	fsx: CreateDataRepositoryTask
	fsx: CreateFileCache
	fsx: CreateFileSystem
	fsx: CreateFileSystemFromBackup
	fsx: CreateSnapshot
	fsx: CreateStorageVirtualMachine
	fsx: CreateVolume
	fsx: CreateVolumeFromBackup
	fsx: DeleteBackup
	fsx: DeleteFileCache
	fsx: DeleteFileSystem
	fsx: DeleteSnapshot
	fsx: DeleteStorageVirtualMachine
	fsx: DeleteVolume
	fsx: DescribeBackups
	fsx: DescribeDataRepositoryAssociations
	fsx: DescribeDataRepositoryTasks
	fsx: DescribeFileCaches

Awalan layanan	Tindakan
	fsx: DescribeFileSystemAliases
	fsx: DescribeFileSystems
	fsx: DescribeS3AccessPointAttachments
	fsx: DescribeSharedVpcConfiguration
	fsx: DescribeSnapshots
	fsx: DescribeStorageVirtualMachines
	fsx: DescribeVolumes
	fsx: DetachAndDeleteS3AccessPoint
	fsx: DisassociateFileSystemAliases
	fsx: ReleaseFileSystemNfsV3Locks
	fsx: RestoreVolumeFromSnapshot
	fsx: StartMisconfiguredStateRecovery
	fsx: UpdateDataRepositoryAssociation
	fsx: UpdateFileCache
	fsx: UpdateFileSystem
	fsx: UpdateSharedVpcConfiguration
	fsx: UpdateSnapshot
	fsx: UpdateStorageVirtualMachine
	fsx: UpdateVolume

Awalan layanan	Tindakan
gamelift	gamelift: AcceptMatch gamelift: ClaimGameServer gamelift: CreateAlias gamelift: CreateBuild gamelift: CreateContainerGroupDefinition gamelift: CreateFleet gamelift: CreateFleetLocations gamelift: CreateGameServerGroup gamelift: CreateGameSession gamelift: CreateGameSessionQueue gamelift: CreateLocation gamelift: CreateMatchmakingConfiguration gamelift: CreateMatchmakingRuleSet gamelift: CreatePlayerSession gamelift: CreatePlayerSessions gamelift: CreateScript gamelift: CreateVpcPeeringAuthorization gamelift: CreateVpcPeeringConnection gamelift: DeleteAlias gamelift: DeleteBuild gamelift: DeleteContainerGroupDefinition

Awalan layanan	Tindakan
	gamelift: DeleteFleet gamelift: DeleteFleetLocations gamelift: DeleteGameServerGroup gamelift: DeleteGameSessionQueue gamelift: DeleteLocation gamelift: DeleteMatchmakingConfiguration gamelift: DeleteMatchmakingRuleSet gamelift: DeleteScalingPolicy gamelift: DeleteScript gamelift: DeleteVpcPeeringAuthorization gamelift: DeleteVpcPeeringConnection gamelift: DeregisterCompute gamelift: DeregisterGameServer gamelift: DescribeAlias gamelift: DescribeBuild gamelift: DescribeCompute gamelift: DescribeContainerFleet gamelift: DescribeContainerGroupDefinition gamelift: DescribeContainerGroupPortMappings gamelift: DescribeEC2InstanceLimits gamelift: DescribeFleetAttributes

Awalan layanan	Tindakan
	gamelift: DescribeFleetCapacity gamelift: DescribeFleetEvents gamelift: DescribeFleetLocationAttributes gamelift: DescribeFleetLocationCapacity gamelift: DescribeFleetLocationUtilization gamelift: DescribeFleetPortSettings gamelift: DescribeFleetUtilization gamelift: DescribeGameServer gamelift: DescribeGameServerGroup gamelift: DescribeGameServerInstances gamelift: DescribeGameSessionDetails gamelift: DescribeGameSessionPlacement gamelift: DescribeGameSessionQueues gamelift: DescribeGameSessions gamelift: DescribeInstances gamelift: DescribeMatchmaking gamelift: DescribeMatchmakingConfigurations gamelift: DescribeMatchmakingRuleSets gamelift: DescribePlayerSessions gamelift: DescribeRuntimeConfiguration gamelift: DescribeScalingPolicies

Awalan layanan	Tindakan
	gamelift: DescribeScript gamelift: DescribeVpcPeeringAuthorizations gamelift: DescribeVpcPeeringConnections gamelift: GetComputeAccess gamelift: GetComputeAuthToken gamelift: GetGameSessionLogUrl gamelift: GetInstanceAccess gamelift: GetPlayerConnectionDetails gamelift: ListAliases gamelift: ListBuilds gamelift: ListCompute gamelift: ListContainerFleets gamelift: ListContainerGroupDefinitionVersions gamelift: ListContainerGroupDefinitions gamelift: ListFleetDeployments gamelift: ListFleets gamelift: ListGameServerGroups gamelift: ListGameServers gamelift: ListLocations gamelift: ListScripts gamelift: PutScalingPolicy

Awalan layanan	Tindakan
	gamelift: RegisterCompute gamelift: RegisterGameServer gamelift: RequestUploadCredentials gamelift: ResolveAlias gamelift: ResumeGameServerGroup gamelift: SearchGameSessions gamelift: StartFleetActions gamelift: StartGameSessionPlacement gamelift: StartMatchBackfill gamelift: StartMatchmaking gamelift: StopFleetActions gamelift: StopGameSessionPlacement gamelift: StopMatchmaking gamelift: SuspendGameServerGroup gamelift: TerminateGameSession gamelift: UpdateAlias gamelift: UpdateBuild gamelift: UpdateContainerGroupDefinition gamelift: UpdateFleetAttributes gamelift: UpdateFleetCapacity gamelift: UpdateFleetPortSettings

Awalan layanan	Tindakan
	gamelift: UpdateGameServer gamelift: UpdateGameServerGroup gamelift: UpdateGameSession gamelift: UpdateGameSessionQueue gamelift: UpdateMatchmakingConfiguration gamelift: UpdateRuntimeConfiguration gamelift: UpdateScript gamelift: ValidateMatchmakingRuleSet

Awalan layanan	Tindakan
geo	geografis: AssociateTrackerConsumer
	geografis: BatchDeleteDevicePositionHistory
	geografis: BatchDeleteGeofence
	geografis: BatchEvaluateGeofences
	geografis: BatchGetDevicePosition
	geografis: BatchPutGeofence
	geografis: BatchUpdateDevicePosition
	geografis: CalculateRoute
	geografis: CalculateRouteMatrix
	geografis: CancelJob
	geografis: CreateGeofenceCollection
	geografis: CreateMap
	geografis: CreatePlaceIndex
	geografis: CreateRouteCalculator
	geografis: CreateTracker
	geografis: DeleteGeofenceCollection
	geografis: DeleteKey
	geografis: DeleteMap
	geografis: DeletePlaceIndex
	geografis: DeleteRouteCalculator
	geografis: DeleteTracker

Awalan layanan	Tindakan
	geografis: DescribeGeofenceCollection
	geografis: DescribeKey
	geografis: DescribeMap
	geografis: DescribePlaceIndex
	geografis: DescribeRouteCalculator
	geografis: DescribeTracker
	geografis: DisassociateTrackerConsumer
	geografis: ForecastGeofenceEvents
	geografis: GetDevicePosition
	geografis: GetDevicePositionHistory
	geografis: GetGeofence
	geografis: GetJob
	geografis: GetMapGlyphs
	geografis: GetMapSprites
	geografis: GetMapStyleDescriptor
	geografis: GetMapTile
	geografis: GetPlace
	geografis: ListDevicePositions
	geografis: ListGeofenceCollections
	geografis: ListGeofences
	geografis: ListJobs

Awalan layanan	Tindakan
	geografis: ListKeys
	geografis: ListMaps
	geografis: ListPlaceIndexes
	geografis: ListRouteCalculators
	geografis: ListTrackerConsumers
	geografis: ListTrackers
	geografis: PutGeofence
	geografis: SearchPlaceIndexForPosition
	geografis: SearchPlaceIndexForSuggestions
	geografis: SearchPlaceIndexForText
	geografis: UpdateGeofenceCollection
	geografis: UpdateKey
	geografis: UpdateMap
	geografis: UpdatePlaceIndex
	geografis: UpdateRouteCalculator
	geografis: UpdateTracker
	geografis: VerifyDevicePosition

Awalan layanan	Tindakan
gletser	gletser: AbortMultipartUpload gletser: AbortVaultLock gletser: CompleteMultipartUpload gletser: CompleteVaultLock gletser: CreateVault gletser: DeleteArchive gletser: DeleteVault gletser: DeleteVaultAccessPolicy gletser: DeleteVaultNotifications gletser: DescribeJob gletser: DescribeVault gletser: GetDataRetrievalPolicy gletser: GetJobOutput gletser: GetVaultAccessPolicy gletser: GetVaultLock gletser: GetVaultNotifications gletser: InitiateJob gletser: InitiateMultipartUpload gletser: InitiateVaultLock gletser: ListJobs gletser: ListMultipartUploads

Awalan layanan	Tindakan
	gletser: ListParts gletser: ListProvisionedCapacity gletser: ListVaults gletser: PurchaseProvisionedCapacity gletser: SetDataRetrievalPolicy gletser: SetVaultAccessPolicy gletser: SetVaultNotifications gletser: UploadArchive gletser: UploadMultipartPart

Awalan layanan	Tindakan
grafana	grafana: AssociateLicense
	grafana: CreateWorkspace
	grafana: CreateWorkspaceApiKey
	grafana: CreateWorkspaceServiceAccount
	grafana: CreateWorkspaceServiceAccountToken
	grafana: DeleteWorkspace
	grafana: DeleteWorkspaceApiKey
	grafana: DeleteWorkspaceServiceAccount
	grafana: DeleteWorkspaceServiceAccountToken
	grafana: DescribeWorkspace
	grafana: DescribeWorkspaceAuthentication
	grafana: DescribeWorkspaceConfiguration
	grafana: DisassociateLicense
	grafana: ListPermissions
	grafana: ListVersions
	grafana: ListWorkspaceServiceAccountTokens
	grafana: ListWorkspaceServiceAccounts
	grafana: ListWorkspaces
	grafana: UpdatePermissions
	grafana: UpdateWorkspace
	grafana: UpdateWorkspaceAuthentication

Awalan layanan	Tindakan
	grafana: UpdateWorkspaceConfiguration

Awalan layanan	Tindakan
greengrass	greengrass: AssociateRoleToGroup greengrass: AssociateServiceRoleToAccount greengrass: BatchAssociateClientDeviceWithCoreDevice greengrass: BatchDisassociateClientDeviceFromCoreDevice greengrass: CancelDeployment greengrass: CreateComponentVersion greengrass: CreateConnectorDefinition greengrass: CreateConnectorDefinitionVersion greengrass: CreateCoreDefinition greengrass: CreateCoreDefinitionVersion greengrass: CreateDeployment greengrass: CreateDeviceDefinition greengrass: CreateDeviceDefinitionVersion greengrass: CreateFunctionDefinition greengrass: CreateFunctionDefinitionVersion greengrass: CreateGroup greengrass: CreateGroupCertificateAuthority greengrass: CreateGroupVersion greengrass: CreateLoggerDefinition greengrass: CreateLoggerDefinitionVersion greengrass: CreateResourceDefinition

Awalan layanan	Tindakan
	greengrass: CreateResourceDefinitionVersion greengrass: CreateSoftwareUpdateJob greengrass: CreateSubscriptionDefinition greengrass: CreateSubscriptionDefinitionVersion greengrass: DeleteComponent greengrass: DeleteConnectorDefinition greengrass: DeleteCoreDefinition greengrass: DeleteCoreDevice greengrass: DeleteDeployment greengrass: DeleteDeviceDefinition greengrass: DeleteFunctionDefinition greengrass: DeleteGroup greengrass: DeleteLoggerDefinition greengrass: DeleteResourceDefinition greengrass: DeleteSubscriptionDefinition greengrass: DescribeComponent greengrass: DisassociateRoleFromGroup greengrass: DisassociateServiceRoleFromAccount greengrass: GetAssociatedRole greengrass: GetBulkDeploymentStatus greengrass: GetComponent

Awalan layanan	Tindakan
	greengrass: GetComponentVersionArtifact greengrass: GetConnectivityInfo greengrass: GetConnectorDefinition greengrass: GetConnectorDefinitionVersion greengrass: GetCoreDefinition greengrass: GetCoreDefinitionVersion greengrass: GetCoreDevice greengrass: GetDeployment greengrass: GetDeploymentStatus greengrass: GetDeviceDefinition greengrass: GetDeviceDefinitionVersion greengrass: GetFunctionDefinition greengrass: GetFunctionDefinitionVersion greengrass: GetGroup greengrass: GetGroupCertificateAuthority greengrass: GetGroupCertificateConfiguration greengrass: GetGroupVersion greengrass: GetLoggerDefinition greengrass: GetLoggerDefinitionVersion greengrass: GetResourceDefinition greengrass: GetResourceDefinitionVersion

Awalan layanan	Tindakan
	greengrass: GetServiceRoleForAccount greengrass: GetSubscriptionDefinition greengrass: GetSubscriptionDefinitionVersion greengrass: GetThingRuntimeConfiguration greengrass: ListBulkDeploymentDetailedReports greengrass: ListBulkDeployments greengrass: ListClientDevicesAssociatedWithCoreDevice greengrass: ListComponentVersions greengrass: ListComponents greengrass: ListConnectorDefinitionVersions greengrass: ListConnectorDefinitions greengrass: ListCoreDefinitionVersions greengrass: ListCoreDefinitions greengrass: ListCoreDevices greengrass: ListDeployments greengrass: ListDeviceDefinitionVersions greengrass: ListDeviceDefinitions greengrass: ListEffectiveDeployments greengrass: ListFunctionDefinitionVersions greengrass: ListFunctionDefinitions greengrass: ListGroupCertificateAuthorities

Awalan layanan	Tindakan
	greengrass: ListGroupVersions greengrass: ListGroups greengrass: ListInstalledComponents greengrass: ListLoggerDefinitionVersions greengrass: ListLoggerDefinitions greengrass: ListResourceDefinitionVersions greengrass: ListResourceDefinitions greengrass: ListSubscriptionDefinitionVersions greengrass: ListSubscriptionDefinitions greengrass: ResetDeployments greengrass: StartBulkDeployment greengrass: StopBulkDeployment greengrass: UpdateConnectivityInfo greengrass: UpdateConnectorDefinition greengrass: UpdateCoreDefinition greengrass: UpdateDeviceDefinition greengrass: UpdateFunctionDefinition greengrass: UpdateGroup greengrass: UpdateGroupCertificateConfiguration greengrass: UpdateLoggerDefinition greengrass: UpdateResourceDefinition

Awalan layanan	Tindakan
	greengrass: UpdateSubscriptionDefinition greengrass: UpdateThingRuntimeConfiguration

Awalan layanan	Tindakan
stasiun tanah	stasiun tanah: CancelContact
	stasiun tanah: CreateConfig
	stasiun tanah: CreateDataflowEndpointGroup
	stasiun tanah: CreateDataflowEndpointGroupV2
	stasiun tanah: CreateEphemeris
	stasiun tanah: CreateMissionProfile
	stasiun tanah: DeleteConfig
	stasiun tanah: DeleteDataflowEndpointGroup
	stasiun tanah: DeleteEphemeris
	stasiun tanah: DeleteMissionProfile
	stasiun tanah: DescribeContact
	stasiun tanah: DescribeContactVersion
	stasiun tanah: DescribeEphemeris
	stasiun tanah: GetConfig
	stasiun tanah: GetDataflowEndpointGroup
	stasiun tanah: GetMinuteUsage
	stasiun tanah: GetMissionProfile
	stasiun tanah: GetSatellite
	stasiun tanah: ListAntennas
	stasiun tanah: ListConfigs
	stasiun tanah: ListContacts

Awalan layanan	Tindakan
	stasiun tanah: ListDataflowEndpointGroups
	stasiun tanah: ListEphemerides
	stasiun tanah: ListGroundStationReservations
	stasiun tanah: ListGroundStations
	stasiun tanah: ListMissionProfiles
	stasiun tanah: ListSatellites
	stasiun tanah: RegisterAgent
	stasiun tanah: ReserveContact
	stasiun tanah: UpdateAgentStatus
	stasiun tanah: UpdateConfig
	stasiun tanah: UpdateEphemeris
	stasiun tanah: UpdateMissionProfile

Awalan layanan	Tindakan
tugas jaga	tugas jaga: AcceptAdministratorInvitation tugas jaga: AcceptInvitation tugas jaga: ArchiveFindings tugas jaga: CreateDetector tugas jaga: CreateFilter GuardDuty:CreateIPset tugas jaga: CreateInvestigation tugas jaga: CreateMalwareProtectionPlan tugas jaga: CreateMembers tugas jaga: CreatePublishingDestination tugas jaga: CreateSampleFindings tugas jaga: CreateThreatEntitySet tugas jaga: CreateThreatIntelSet tugas jaga: CreateTrustedEntitySet tugas jaga: DeclineInvitations tugas jaga: DeleteDetector tugas jaga: DeleteFilter guardDuty:deleteIPset tugas jaga: DeleteInvitations tugas jaga: DeleteMalwareProtectionPlan tugas jaga: DeleteMembers

Awalan layanan	Tindakan
	tugas jaga: DeletePublishingDestination
	tugas jaga: DeleteThreatEntitySet
	tugas jaga: DeleteThreatIntelSet
	tugas jaga: DeleteTrustedEntitySet
	tugas jaga: DescribeMalwareScans
	tugas jaga: DescribeOrganizationConfiguration
	tugas jaga: DescribePublishingDestination
	tugas jaga: DisableOrganizationAdminAccount
	tugas jaga: DisassociateFromAdministratorAccount
	tugas jaga: DisassociateFromMasterAccount
	tugas jaga: DisassociateMembers
	tugas jaga: EnableOrganizationAdminAccount
	tugas jaga: GetAdministratorAccount
	tugas jaga: GetCoverageStatistics
	tugas jaga: GetDetector
	tugas jaga: GetFilter
	tugas jaga: GetFindings
	tugas jaga: GetFindingsStatistics
	GuardDuty:GETIPSET
	tugas jaga: GetInvestigation
	tugas jaga: GetInvitationsCount

Awalan layanan	Tindakan
	tugas jaga: GetMalwareProtectionPlan tugas jaga: GetMalwareScan tugas jaga: GetMalwareScanSettings tugas jaga: GetMasterAccount tugas jaga: GetMemberDetectors tugas jaga: GetMembers tugas jaga: GetOrganizationStatistics tugas jaga: GetRemainingFreeTrialDays tugas jaga: GetThreatEntitySet tugas jaga: GetThreatIntelSet tugas jaga: GetTrustedEntitySet tugas jaga: GetUsageStatistics tugas jaga: InviteMembers tugas jaga: ListCoverage tugas jaga: ListDetectors tugas jaga: ListFilters tugas jaga: ListFindings guardduty:Listipsets tugas jaga: ListInvestigations tugas jaga: ListInvitations tugas jaga: ListMalwareProtectionPlans

Awalan layanan	Tindakan
	tugas jaga: ListMalwareScans tugas jaga: ListMembers tugas jaga: ListOrganizationAdminAccounts tugas jaga: ListPublishingDestinations tugas jaga: ListThreatEntitySets tugas jaga: ListThreatIntelSets tugas jaga: ListTrustedEntitySets tugas jaga: StartMalwareScan tugas jaga: StartMonitoringMembers tugas jaga: StopMonitoringMembers tugas jaga: UnarchiveFindings tugas jaga: UpdateDetector tugas jaga: UpdateFilter tugas jaga: UpdateFindingsFeedback GuardDuty: UpdateIPSET tugas jaga: UpdateMalwareProtectionPlan tugas jaga: UpdateMalwareScanSettings tugas jaga: UpdateMemberDetectors tugas jaga: UpdateOrganizationConfiguration tugas jaga: UpdatePublishingDestination tugas jaga: UpdateThreatEntitySet

Awalan layanan	Tindakan
	tugas jaga: UpdateThreatIntelSet tugas jaga: UpdateTrustedEntitySet

Awalan layanan	Tindakan
healthlake	Healthlake: CancelFHIRExportJobWithDelete
	Healthlake: CreateDataTransformationProfile
	Healthlake:CreateFhirDataStore
	Healthlake: CreateResource
	Healthlake: DeleteDataTransformationProfile
	HealthLake:DeleteFhirDataStore
	Healthlake: DeleteResource
	Healthlake: DescribeDataTransformationJob
	Healthlake:DescribefhirDataStore
	Healthlake: DescribeFHIRExportJob
	Healthlake: DescribeFHIRExportJobWithGet
	Healthlake: DescribeFHIRImportJob
	Healthlake: GetCapabilities
	Healthlake: GetDataTransformationProfile
	Healthlake: ListDataTransformationJobs
	Healthlake: ListDataTransformationProfileVersions
	Healthlake: ListDataTransformationProfiles
	HealthLake:ListfHirDataStore
	Healthlake: ListFHIRExportJobs
	Healthlake: ListFHIRImportJobs
	Healthlake: PublishDataTransformationProfile

Awalan layanan	Tindakan
	Healthlake: ReadResource
	Healthlake: SearchEverything
	Healthlake: SearchWithGet
	Healthlake: SearchWithPost
	Healthlake: StartFHIRExportJob
	Healthlake: StartFHIRExportJobWithPost
	Healthlake: StartFHIRImportJob
	Healthlake: UpdateDataTransformationProfile
	HealthLake:UpdateFhirDataStore
	Healthlake: UpdateProfileWithAgent
	Healthlake: UpdateResource

Awalan layanan	Tindakan
kode madu	kode madu: BatchCreateTableRows kode madu: BatchDeleteTableRows kode madu: BatchUpdateTableRows kode madu: BatchUpsertTableRows kode madu: DescribeTableDataImportJob kode madu: GetScreenData kode madu: InvokeScreenAutomation kode madu: ListTableColumns kode madu: ListTableRows kode madu: ListTables kode madu: QueryTableRows kode madu: StartTableDataImportJob

Awalan layanan	Tindakan
iam	saya: AddClientIDToOpenIDConnectProvider
	saya: AddRoleToInstanceProfile
	saya: AddUserToGroup
	saya: AttachGroupPolicy
	saya: AttachRolePolicy
	saya: AttachUserPolicy
	saya: ChangePassword
	saya: CreateAccessKey
	saya: CreateAccountAlias
	saya: CreateGroup
	saya: CreateInstanceProfile
	saya: CreateLoginProfile
	saya: CreateOpenIDConnectProvider
	saya: CreatePolicy
	saya: CreatePolicyVersion
	saya: CreateRole
	iam:CreateSamlProvider
	saya: CreateServiceLinkedRole
	saya: CreateServiceSpecificCredential
	saya: CreateUser
	saya: CreateVirtualMFADevice

Awalan layanan	Tindakan
	iam:Nonaktifkan FADEVICE
	saya: DeleteAccessKey
	saya: DeleteAccountAlias
	saya: DeleteAccountPasswordPolicy
	saya: DeleteCloudFrontPublicKey
	saya: DeleteGroup
	saya: DeleteGroupPolicy
	saya: DeleteInstanceProfile
	saya: DeleteLoginProfile
	saya: DeleteOpenIDConnectProvider
	saya: DeletePolicy
	saya: DeletePolicyVersion
	saya: DeleteRole
	saya: DeleteRolePermissionsBoundary
	saya: DeleteRolePolicy
	Iam:deleteCamlProvider
	saya: DeleteSSHPublicKey
	saya: DeleteServerCertificate
	saya: DeleteServiceLinkedRole
	saya: DeleteServiceSpecificCredential
	saya: DeleteSigningCertificate

Awalan layanan	Tindakan
	saya: DeleteUser
	saya: DeleteUserPermissionsBoundary
	saya: DeleteUserPolicy
	saya: DeleteVirtualMFADevice
	saya: DetachGroupPolicy
	saya: DetachRolePolicy
	saya: DetachUserPolicy
	saya: DisableOrganizationsRootCredentialsManagement
	saya: DisableOrganizationsRootSessions
	saya: DisableOutboundWebIdentityFederation
	Iam:enableMfadevice
	saya: EnableOrganizationsRootCredentialsManagement
	saya: EnableOrganizationsRootSessions
	saya: EnableOutboundWebIdentityFederation
	saya: GenerateCredentialReport
	saya: GenerateOrganizationsAccessReport
	saya: GenerateServiceLastAccessedDetails
	saya: GetAccessKeyLastUsed
	saya: GetAccountAuthorizationDetails
	saya: GetAccountEmailAddress
	saya: GetAccountName

Awalan layanan	Tindakan
	saya: GetAccountPasswordPolicy
	saya: GetAccountSummary
	saya: GetCloudFrontPublicKey
	saya: GetContextKeysForCustomPolicy
	saya: GetContextKeysForPrincipalPolicy
	saya: GetCredentialReport
	saya: GetGroup
	saya: GetGroupPolicy
	saya: GetInstanceProfile
	saya: GetLoginProfile
	Iam:GetMFADevice
	saya: GetOpenIDConnectProvider
	saya: GetOrganizationsAccessReport
	saya: GetOutboundWebIdentityFederationInfo
	saya: GetPolicy
	saya: GetPolicyVersion
	saya: GetRole
	saya: GetRolePolicy
	iam:GetSamlProvider
	saya: GetSSHPublicKey
	saya: GetServerCertificate

Awalan layanan	Tindakan
	saya: GetServiceLastAccessedDetails
	saya: GetServiceLastAccessedDetailsWithEntities
	saya: GetServiceLinkedRoleDeletionStatus
	saya: GetUser
	saya: GetUserPolicy
	saya: ListAccessKeys
	saya: ListAccountAliases
	saya: ListAttachedGroupPolicies
	saya: ListAttachedRolePolicies
	saya: ListAttachedUserPolicies
	saya: ListCloudFrontPublicKeys
	saya: ListDelegationRequests
	saya: ListEntitiesForPolicy
	saya: ListGroupPolicies
	saya: ListGroups
	saya: ListGroupsForUser
	saya: ListInstanceProfiles
	saya: ListInstanceProfilesForRole
	Iam:ListmFADevices
	saya: ListOpenIDConnectProviders
	saya: ListOrganizationsFeatures

Awalan layanan	Tindakan
	saya: ListPolicies
	saya: ListPoliciesGrantingServiceAccess
	saya: ListPolicyVersions
	saya: ListRolePolicies
	saya: ListRoles
	iam:ListSamlProviders
	saya: ListSSHPublicKeys
	saya: ListSTSRegionalEndpointsStatus
	saya: ListServerCertificates
	saya: ListServiceSpecificCredentials
	saya: ListSigningCertificates
	saya: ListUserPolicies
	saya: ListUsers
	saya: ListVirtualMFADevices
	saya: PutGroupPolicy
	saya: PutRolePermissionsBoundary
	saya: PutRolePolicy
	saya: PutUserPermissionsBoundary
	saya: PutUserPolicy
	saya: RemoveClientIDFromOpenIDConnectProvider
	saya: RemoveRoleFromInstanceProfile

Awalan layanan	Tindakan
	saya: RemoveUserFromGroup
	saya: ResetServiceSpecificCredential
	iam:resyncmfadevice
	saya: SetDefaultPolicyVersion
	saya: SetSTSRegionalEndpointStatus
	saya: SetSecurityTokenServicePreferences
	saya: SimulateCustomPolicy
	saya: SimulatePrincipalPolicy
	saya: UpdateAccessKey
	saya: UpdateAccountEmailAddress
	saya: UpdateAccountName
	saya: UpdateAccountPasswordPolicy
	saya: UpdateAssumeRolePolicy
	saya: UpdateCloudFrontPublicKey
	saya: UpdateGroup
	saya: UpdateLoginProfile
	saya: UpdateOpenIDConnectProviderThumbprint
	saya: UpdateRole
	saya: UpdateRoleDescription
	iam:UpdateSamlProvider
	saya: UpdateSSHPublicKey

Awalan layanan	Tindakan
	saya: UpdateServerCertificate saya: UpdateServiceSpecificCredential saya: UpdateSigningCertificate saya: UpdateUser saya: UploadCloudFrontPublicKey saya: UploadSSHPublicKey saya: UploadServerCertificate saya: UploadSigningCertificate

Awalan layanan	Tindakan
toko identitas	toko identitas: CreateGroup toko identitas: CreateGroupMembership toko identitas: CreateUser toko identitas: DeleteGroup toko identitas: DeleteGroupMembership toko identitas: DeleteUser toko identitas: DescribeGroup toko identitas: DescribeGroupMembership toko identitas: DescribeUser toko identitas: GetGroupId toko identitas: GetGroupMembershipId toko identitas: GetUserId toko identitas: IsMemberInGroups toko identitas: ListGroupMemberships toko identitas: ListGroupMembershipsForMember toko identitas: ListGroups toko identitas: ListUsers toko identitas: UpdateGroup toko identitas: UpdateUser

Awalan layanan	Tindakan
imagebuilder	imagebuilder: CancelImageCreation
	imagebuilder: CancelLifecycleExecution
	imagebuilder: CreateComponent
	imagebuilder: CreateContainerRecipe
	imagebuilder: CreateDistributionConfiguration
	imagebuilder: CreateImage
	imagebuilder: CreateImagePipeline
	imagebuilder: CreateImageRecipe
	imagebuilder: CreateInfrastructureConfiguration
	imagebuilder: CreateLifecyclePolicy
	imagebuilder: CreateWorkflow
	imagebuilder: DeleteComponent
	imagebuilder: DeleteContainerRecipe
	imagebuilder: DeleteDistributionConfiguration
	imagebuilder: DeleteImage
	imagebuilder: DeleteImagePipeline
	imagebuilder: DeleteImageRecipe
	imagebuilder: DeleteInfrastructureConfiguration
	imagebuilder: DeleteLifecyclePolicy
	imagebuilder: DeleteWorkflow
	imagebuilder: DistributeImage

Awalan layanan	Tindakan
	imagebuilder: GetComponentPolicy
	imagebuilder: GetContainerRecipePolicy
	imagebuilder: GetImagePolicy
	imagebuilder: GetImageRecipePolicy
	imagebuilder: GetLifecycleExecution
	imagebuilder: GetLifecyclePolicy
	imagebuilder: GetMarketplaceResource
	imagebuilder: GetWorkflowExecution
	imagebuilder: GetWorkflowStepExecution
	imagebuilder: ImportComponent
	imagebuilder: ImportDiskImage
	imagebuilder: ImportVmImage
	imagebuilder: ListComponentBuildVersions
	imagebuilder: ListComponents
	imagebuilder: ListContainerRecipes
	imagebuilder: ListDistributionConfigurations
	imagebuilder: ListImageBuildVersions
	imagebuilder: ListImagePackages
	imagebuilder: ListImagePipelineImages
	imagebuilder: ListImagePipelines
	imagebuilder: ListImageRecipes

Awalan layanan	Tindakan
	imagebuilder: ListImageScanFindingAggregations
	imagebuilder: ListImageScanFindings
	imagebuilder: ListImages
	imagebuilder: ListInfrastructureConfigurations
	imagebuilder: ListLifecycleExecutionResources
	imagebuilder: ListLifecycleExecutions
	imagebuilder: ListLifecyclePolicies
	imagebuilder: ListWaitingWorkflowSteps
	imagebuilder: ListWorkflowExecutions
	imagebuilder: ListWorkflowStepExecutions
	imagebuilder: ListWorkflows
	imagebuilder: PutComponentPolicy
	imagebuilder: PutContainerRecipePolicy
	imagebuilder: PutImagePolicy
	imagebuilder: PutImageRecipePolicy
	imagebuilder: RetryImage
	imagebuilder: SendWorkflowStepAction
	imagebuilder: StartImagePipelineExecution
	imagebuilder: StartResourceStateUpdate
	imagebuilder: UpdateDistributionConfiguration
	imagebuilder: UpdateImagePipeline

Awalan layanan	Tindakan
	imagebuilder: UpdateInfrastructureConfiguration

Awalan layanan	Tindakan
inspektur	inspektur: AddAttributesToFindings inspektur: CreateAssessmentTarget inspektur: CreateAssessmentTemplate inspektur: CreateExclusionsPreview inspektur: CreateResourceGroup inspektur: DeleteAssessmentRun inspektur: DeleteAssessmentTarget inspektur: DeleteAssessmentTemplate inspektur: DescribeAssessmentRuns inspektur: DescribeAssessmentTargets inspektur: DescribeAssessmentTemplates inspektur: DescribeCrossAccountAccessRole inspektur: DescribeExclusions inspektur: DescribeFindings inspektur: DescribeResourceGroups inspektur: DescribeRulesPackages inspektur: GetAssessmentReport inspektur: GetExclusionsPreview inspektur: GetTelemetryMetadata inspektur: ListAssessmentRunAgents inspektur: ListAssessmentRuns

Awalan layanan	Tindakan
	inspektur: ListAssessmentTargets inspektur: ListAssessmentTemplates inspektur: ListEventSubscriptions inspektur: ListExclusions inspektur: ListFindings inspektur: ListRulesPackages inspektur: PreviewAgents inspektur: RegisterCrossAccountAccessRole inspektur: RemoveAttributesFromFindings inspektur: StartAssessmentRun inspektur: StopAssessmentRun inspektur: SubscribeToEvent inspektur: UnsubscribeFromEvent inspektur: UpdateAssessmentTarget

Awalan layanan	Tindakan
inspektor2	inspektor2: AssociateMember
	inspektor2: BatchGetAccountStatus
	inspektor2: BatchGetCodeSnippet
	inspektor2: BatchGetFindingDetails
	inspektor2: BatchGetFreeTrialInfo
	inspektor2: BatchGetMemberEc2DeepInspectionStatus
	inspektor2: BatchUpdateMemberEc2DeepInspectionStatus
	inspektor2: CancelFindingsReport
	inspektor2: CancelSbomExport
	inspektor2: CreateCisScanConfiguration
	inspektor2: CreateCodeSecurityIntegration
	inspektor2: CreateConnector
	inspektor2: CreateFilter
	inspektor2: CreateFindingsReport
	inspektor2: CreateSbomExport
	inspektor2: DeleteCisScanConfiguration
	inspektor2: DeleteCodeSecurityIntegration
	inspektor2: DeleteConnector
	inspektor2: DeleteFilter
	inspektor2: DescribeOrganizationConfiguration
	Inspector2: Nonaktifkan

Awalan layanan	Tindakan
	inspektor2: DisableDelegatedAdminAccount
	inspektor2: DisassociateMember
	Inspector2: aktifkan
	inspektor2: EnableDelegatedAdminAccount
	inspektor2: GetCisScanReport
	inspektor2: GetCisScanResultDetails
	inspektor2: GetClustersForImage
	inspektor2: GetCodeSecurityIntegration
	inspektor2: GetCodeSecurityScan
	inspektor2: GetConfiguration
	inspektor2: GetDelegatedAdminAccount
	inspektor2: GetEc2DeepInspectionConfiguration
	inspektor2: GetEncryptionKey
	inspektor2: GetFindingsReportStatus
	inspektor2: GetMember
	inspektor2: GetSbomExport
	inspektor2: ListAccountPermissions
	inspektor2: ListCisScanConfigurations
	inspektor2: ListCisScanResultsAggregatedByChecks
	inspektor2: ListCisScanResultsAggregatedByTargetResource
	inspektor2: ListCisScans

Awalan layanan	Tindakan
	inspektor2: ListCodeSecurityIntegrations
	inspektor2: ListCodeSecurityScanConfigurations
	inspektor2: ListConnectorScanConfigurations
	inspektor2: ListConnectors
	inspektor2: ListCoverage
	inspektor2: ListCoverageStatistics
	inspektor2: ListDelegatedAdminAccounts
	inspektor2: ListFilters
	inspektor2: ListFindingAggregations
	inspektor2: ListFindings
	inspektor2: ListMembers
	inspektor2: ListUsageTotals
	inspektor2: ResetEncryptionKey
	inspektor2: SearchVulnerabilities
	inspektor2: SendCisSessionHealth
	inspektor2: SendCisSessionTelemetry
	inspektor2: StartCisSession
	inspektor2: StartCodeSecurityScan
	inspektor2: StopCisSession
	inspektor2: UpdateCisScanConfiguration
	inspektor2: UpdateCodeSecurityIntegration

Awalan layanan	Tindakan
	inspektor2: UpdateConfiguration
	inspektor2: UpdateConnector
	inspektor2: UpdateEc2DeepInspectionConfiguration
	inspektor2: UpdateEncryptionKey
	inspektor2: UpdateFilter
	inspektor2: UpdateOrgEc2DeepInspectionConfiguration
	inspektor2: UpdateOrganizationConfiguration

Awalan layanan	Tindakan
iot	IOT: AcceptCertificateTransfer IOT: AddThingToBillingGroup IOT: AddThingToThingGroup IOT: AssociateSbomWithPackageVersion IOT: AssociateTargetsWithJob IOT: AttachPolicy IOT: AttachPrincipalPolicy IOT: AttachSecurityProfile IOT: AttachThingPrincipal IOT: CancelAuditMitigationActionsTask IOT: CancelAuditTask IOT: CancelCertificateTransfer IOT: CancelDetectMitigationActionsTask IOT: CancelJob IOT: CancelJobExecution IOT: ClearDefaultAuthorizer IOT: ConfirmTopicRuleDestination IOT: CreateAuditSuppression IOT: CreateAuthorizer IOT: CreateBillingGroup IOT: CreateCertificateFromCsr

Awalan layanan	Tindakan
	IOT: CreateCertificateProvider IOT: CreateCommand IOT: CreateCustomMetric IOT: CreateDimension IOT: CreateDomainConfiguration IOT: CreateDynamicThingGroup IOT: CreateFleetMetric IOT: CreateJob IOT: CreateJobTemplate IOT: CreateKeysAndCertificate IOT: CreateMitigationAction IoT: CreateoTaupdate IOT: CreatePackage IOT: CreatePackageVersion IOT: CreatePolicy IOT: CreatePolicyVersion IOT: CreateProvisioningClaim IOT: CreateProvisioningTemplate IOT: CreateProvisioningTemplateVersion IOT: CreateRoleAlias IOT: CreateScheduledAudit

Awalan layanan	Tindakan
	IOT: CreateSecurityProfile IOT: CreateStream IOT: CreateThing IOT: CreateThingGroup IOT: CreateThingType IOT: CreateTopicRule IOT: CreateTopicRuleDestination IOT: DeleteAccountAuditConfiguration IOT: DeleteAuditSuppression IOT: DeleteAuthorizer IOT: DeleteBillingGroup IoT: DeleteCacertificate IOT: DeleteCertificate IOT: DeleteCertificateProvider IOT: DeleteCommand IOT: DeleteCustomMetric IOT: DeleteDimension IOT: DeleteDomainConfiguration IOT: DeleteDynamicThingGroup IOT: DeleteFleetMetric IOT: DeleteJob

Awalan layanan	Tindakan
	IOT: DeleteJobExecution IOT: DeleteJobTemplate IOT: DeleteMitigationAction IoT: DeleteOtaUpdate IOT: DeletePackage IOT: DeletePackageVersion IOT: DeletePolicy IOT: DeletePolicyVersion IOT: DeleteProvisioningTemplate IOT: DeleteProvisioningTemplateVersion IOT: DeleteRegistrationCode IOT: DeleteRoleAlias IOT: DeleteScheduledAudit IOT: DeleteSecurityProfile IOT: DeleteStream IOT: DeleteThing IOT: DeleteThingGroup IOT: DeleteThingType IOT: DeleteTopicRule IOT: DeleteTopicRuleDestination IOT: DeleteV2LoggingLevel

Awalan layanan	Tindakan
	IOT: DeprecateThingType
	IOT: DescribeAccountAuditConfiguration
	IOT: DescribeAuditFinding
	IOT: DescribeAuditMitigationActionsTask
	IOT: DescribeAuditSuppression
	IOT: DescribeAuditTask
	IOT: DescribeAuthorizer
	IOT: DescribeBillingGroup
	IoT:Describecacertificate
	IOT: DescribeCertificate
	IOT: DescribeCertificateProvider
	IOT: DescribeCustomMetric
	IOT: DescribeDefaultAuthorizer
	IOT: DescribeDetectMitigationActionsTask
	IOT: DescribeDimension
	IOT: DescribeDomainConfiguration
	IOT: DescribeEncryptionConfiguration
	IOT: DescribeEndpoint
	IOT: DescribeEventConfigurations
	IOT: DescribeFleetMetric
	IOT: DescribeIndex

Awalan layanan	Tindakan
	IOT: DescribeJob IOT: DescribeJobExecution IOT: DescribeJobTemplate IOT: DescribeManagedJobTemplate IOT: DescribeMitigationAction IOT: DescribeProvisioningTemplate IOT: DescribeProvisioningTemplateVersion IOT: DescribeRoleAlias IOT: DescribeScheduledAudit IOT: DescribeSecurityProfile IOT: DescribeStream IOT: DescribeThing IOT: DescribeThingGroup IOT: DescribeThingRegistrationTask IOT: DescribeThingType IOT: DetachPolicy IOT: DetachPrincipalPolicy IOT: DetachSecurityProfile IOT: DetachThingPrincipal IOT: DisableTopicRule IOT: DisassociateSbomFromPackageVersion

Awalan layanan	Tindakan
	IOT: EnableTopicRule
	IOT: GetBehaviorModelTrainingSummaries
	IOT: GetBucketsAggregation
	IOT: GetCardinality
	IOT: GetCommand
	IOT: GetEffectivePolicies
	IOT: GetJobDocument
	IOT: GetLoggingOptions
	IoT: getoTaupdate
	IOT: GetPackage
	IOT: GetPackageConfiguration
	IOT: GetPackageVersion
	IOT: GetPercentiles
	IOT: GetPolicy
	IOT: GetPolicyVersion
	IOT: GetRegistrationCode
	IOT: GetStatistics
	IOT: GetThingConnectivityData
	IOT: GetTopicRule
	IOT: GetTopicRuleDestination
	IOT: GetV2LoggingOptions

Awalan layanan	Tindakan
	IOT: ListActiveViolations IOT: ListAttachedPolicies IOT: ListAuditFindings IOT: ListAuditMitigationActionsExecutions IOT: ListAuditMitigationActionsTasks IOT: ListAuditSuppressions IOT: ListAuditTasks IOT: ListAuthorizers IOT: ListBillingGroups IoT: ListCacertificates IOT: ListCertificateProviders IOT: ListCertificates IOT: ListCertificatesByCA IOT: ListCommands IOT: ListCustomMetrics IOT: ListDetectMitigationActionsExecutions IOT: ListDetectMitigationActionsTasks IOT: ListDimensions IOT: ListDomainConfigurations IOT: ListFleetMetrics IOT: ListIndices

Awalan layanan	Tindakan
	IOT: ListJobExecutionsForJob IOT: ListJobExecutionsForThing IOT: ListJobTemplates IOT: ListJobs IOT: ListManagedJobTemplates IOT: ListMetricValues IOT: ListMitigationActions IoT: ListoTaupdates IOT: ListOutgoingCertificates IOT: ListPackageVersions IOT: ListPackages IOT: ListPolicies IOT: ListPolicyPrincipals IOT: ListPolicyVersions IOT: ListPrincipalPolicies IOT: ListPrincipalThings IOT: ListPrincipalThingsV2 IOT: ListProvisioningTemplateVersions IOT: ListProvisioningTemplates IOT: ListRelatedResourcesForAuditFinding IOT: ListRoleAliases

Awalan layanan	Tindakan
	IOT: ListSbomValidationResults
	IOT: ListScheduledAudits
	IOT: ListSecurityProfiles
	IOT: ListSecurityProfilesForTarget
	IOT: ListStreams
	IOT: ListTargetsForPolicy
	IOT: ListTargetsForSecurityProfile
	IOT: ListThingGroups
	IOT: ListThingGroupsForThing
	IOT: ListThingPrincipals
	IOT: ListThingPrincipalsV2
	IOT: ListThingRegistrationTaskReports
	IOT: ListThingRegistrationTasks
	IOT: ListThingTypes
	IOT: ListThings
	IOT: ListThingsInBillingGroup
	IOT: ListThingsInThingGroup
	IOT: ListTopicRuleDestinations
	IOT: ListTopicRules
	IOT: ListV2LoggingLevels
	IOT: ListViolationEvents

Awalan layanan	Tindakan
	IOT: PutVerificationStateOnViolation
	IoT: registercacertificate
	IOT: RegisterCertificate
	IOT: RegisterCertificateWithoutCA
	IOT: RegisterThing
	IOT: RejectCertificateTransfer
	IOT: RemoveThingFromBillingGroup
	IOT: RemoveThingFromThingGroup
	IOT: ReplaceTopicRule
	IOT: SearchIndex
	IOT: SetDefaultAuthorizer
	IOT: SetDefaultPolicyVersion
	IOT: SetLoggingOptions
	IOT: SetV2LoggingLevel
	IOT: SetV2LoggingOptions
	IOT: StartAuditMitigationActionsTask
	IOT: StartDetectMitigationActionsTask
	IOT: StartOnDemandAuditTask
	IOT: StartThingRegistrationTask
	IOT: StopThingRegistrationTask
	IOT: TestAuthorization

Awalan layanan	Tindakan
	IOT: TestInvokeAuthorizer IOT: TransferCertificate IOT: UpdateAccountAuditConfiguration IOT: UpdateAuditSuppression IOT: UpdateAuthorizer IOT: UpdateBillingGroup IoT: UpdateCacertificate IOT: UpdateCertificate IOT: UpdateCertificateProvider IOT: UpdateCommand IOT: UpdateCustomMetric IOT: UpdateDimension IOT: UpdateDomainConfiguration IOT: UpdateDynamicThingGroup IOT: UpdateEncryptionConfiguration IOT: UpdateEventConfigurations IOT: UpdateFleetMetric IOT: UpdateIndexingConfiguration IOT: UpdateJob IOT: UpdateMitigationAction IOT: UpdatePackage

Awalan layanan	Tindakan
	IOT: UpdatePackageConfiguration IOT: UpdatePackageVersion IOT: UpdateProvisioningTemplate IOT: UpdateRoleAlias IOT: UpdateScheduledAudit IOT: UpdateSecurityProfile IOT: UpdateStream IOT: UpdateThing IOT: UpdateThingGroup IOT: UpdateThingGroupsForThing IOT: UpdateThingType IOT: UpdateTopicRuleDestination IOT: ValidateSecurityProfileBehaviors

Awalan layanan	Tindakan
iotanalitik	iotanalitik: CancelPipelineReprocessing
	iotanalitik: CreateChannel
	iotanalitik: CreateDataset
	iotanalitik: CreateDatasetContent
	iotanalitik: CreateDatastore
	iotanalitik: CreatePipeline
	iotanalitik: DeleteChannel
	iotanalitik: DeleteDataset
	iotanalitik: DeleteDatasetContent
	iotanalitik: DeleteDatastore
	iotanalitik: DeletePipeline
	iotanalitik: DescribeChannel
	iotanalitik: DescribeDataset
	iotanalitik: DescribeDatastore
	iotanalitik: DescribeLoggingOptions
	iotanalitik: DescribePipeline
	iotanalitik: GetDatasetContent
	iotanalitik: ListChannels
	iotanalitik: ListDatasetContents
	iotanalitik: ListDatasets
	iotanalitik: ListDatastores

Awalan layanan	Tindakan
	iotanalitik: ListPipelines iotanalitik: PutLoggingOptions iotanalitik: RunPipelineActivity iotanalitik: SampleChannelData iotanalitik: StartPipelineReprocessing iotanalitik: UpdateChannel iotanalitik: UpdateDataset iotanalitik: UpdateDatastore iotanalitik: UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor: CreateSuiteDefinition iotdeviceadvisor: DeleteSuiteDefinition iotdeviceadvisor: GetEndpoint iotdeviceadvisor: GetSuiteDefinition iotdeviceadvisor: GetSuiteRun iotdeviceadvisor: GetSuiteRunReport iotdeviceadvisor: ListSuiteDefinitions iotdeviceadvisor: ListSuiteRuns iotdeviceadvisor: StartSuiteRun iotdeviceadvisor: StopSuiteRun iotdeviceadvisor: UpdateSuiteDefinition

Awalan layanan	Tindakan
iotevents	iotevents: BatchAcknowledgeAlarm
	iotevents: BatchDeleteDetector
	iotevents: BatchDisableAlarm
	iotevents: BatchEnableAlarm
	iotevents: BatchResetAlarm
	iotevents: BatchSnoozeAlarm
	iotevents: BatchUpdateDetector
	iotevents: CreateAlarmModel
	iotevents: CreateDetectorModel
	iotevents: CreateInput
	iotevents: DeleteAlarmModel
	iotevents: DeleteDetectorModel
	iotevents: DeleteInput
	iotevents: DescribeAlarm
	iotevents: DescribeAlarmModel
	iotevents: DescribeDetector
	iotevents: DescribeDetectorModel
	iotevents: DescribeDetectorModelAnalysis
	iotevents: DescribeInput
	iotevents: DescribeLoggingOptions
	iotevents: GetDetectorModelAnalysisResults

Awalan layanan	Tindakan
	iotevents: ListAlarmModelVersions iotevents: ListAlarmModels iotevents: ListAlarms iotevents: ListDetectorModelVersions iotevents: ListDetectorModels iotevents: ListDetectors iotevents: ListInputRoutings iotevents: ListInputs iotevents: PutLoggingOptions iotevents: StartDetectorModelAnalysis iotevents: UpdateAlarmModel iotevents: UpdateDetectorModel iotevents: UpdateInput
iotfleethub	iotfleethub: CreateApplication iotfleethub: DeleteApplication iotfleethub: DescribeApplication iotfleethub: ListApplications iotfleethub: UpdateApplication

Awalan layanan	Tindakan
iotsitewise	iotsitewise: AssociateAssets
	iotsitewise: AssociateTimeSeriesToAssetProperty
	iotsitewise: BatchAssociateProjectAssets
	iotsitewise: BatchDisassociateProjectAssets
	iotsitewise: CreateAccessPolicy
	iotsitewise: CreateAsset
	iotsitewise: CreateAssetModel
	iotsitewise: CreateAssetModelCompositeModel
	iotsitewise: CreateBulkImportJob
	iotsitewise: CreateComputationModel
	iotsitewise: CreateDashboard
	iotsitewise: CreateDataset
	iotsitewise: CreateGateway
	iotsitewise: CreatePortal
	iotsitewise: CreateProject
	iotsitewise: DeleteAccessPolicy
	iotsitewise: DeleteAsset
	iotsitewise: DeleteAssetModel
	iotsitewise: DeleteAssetModelCompositeModel
	iotsitewise: DeleteComputationModel
	iotsitewise: DeleteDashboard

Awalan layanan	Tindakan
	iotsitewise: DeleteDataset
	iotsitewise: DeleteGateway
	iotsitewise: DeletePortal
	iotsitewise: DeleteProject
	iotsitewise: DeleteTimeSeries
	iotsitewise: DescribeAccessPolicy
	iotsitewise: DescribeAsset
	iotsitewise: DescribeAssetCompositeModel
	iotsitewise: DescribeAssetModel
	iotsitewise: DescribeAssetModelCompositeModel
	iotsitewise: DescribeAssetModelInterfaceRelationship
	iotsitewise: DescribeAssetProperty
	iotsitewise: DescribeBulkImportJob
	iotsitewise: DescribeComputationModel
	iotsitewise: DescribeComputationModelExecutionSummary
	iotsitewise: DescribeDashboard
	iotsitewise: DescribeDataset
	iotsitewise: DescribeDefaultEncryptionConfiguration
	iotsitewise: DescribeExecution
	iotsitewise: DescribeGateway
	iotsitewise: DescribeGatewayCapabilityConfiguration

Awalan layanan	Tindakan
	iotsitewise: DescribeLoggingOptions
	iotsitewise: DescribePortal
	iotsitewise: DescribeProject
	iotsitewise: DescribeStorageConfiguration
	iotsitewise: DescribeTimeSeries
	iotsitewise: DisassociateAssets
	iotsitewise: DisassociateTimeSeriesFromAssetProperty
	iotsitewise: ExecuteAction
	iotsitewise: ExecuteQuery
	iotsitewise: ListAccessPolicies
	iotsitewise: ListActions
	iotsitewise: ListAssetModelCompositeModels
	iotsitewise: ListAssetModelProperties
	iotsitewise: ListAssetModels
	iotsitewise: ListAssetProperties
	iotsitewise: ListAssetRelationships
	iotsitewise: ListAssets
	iotsitewise: ListAssociatedAssets
	iotsitewise: ListBulkImportJobs
	iotsitewise: ListCompositionRelationships
	iotsitewise: ListComputationModelDataBindingUsages

Awalan layanan	Tindakan
	iotsitewise: ListComputationModelResolveToResources
	iotsitewise: ListComputationModels
	iotsitewise: ListDashboards
	iotsitewise: ListDatasets
	iotsitewise: ListExecutions
	iotsitewise: ListGateways
	iotsitewise: ListInterfaceRelationships
	iotsitewise: ListPortals
	iotsitewise: ListProjectAssets
	iotsitewise: ListProjects
	iotsitewise: ListTimeSeries
	iotsitewise: PutDefaultEncryptionConfiguration
	iotsitewise: PutLoggingOptions
	iotsitewise: PutStorageConfiguration
	iotsitewise: UpdateAccessPolicy
	iotsitewise: UpdateAsset
	iotsitewise: UpdateAssetModel
	iotsitewise: UpdateAssetModelCompositeModel
	iotsitewise: UpdateAssetProperty
	iotsitewise: UpdateComputationModel
	iotsitewise: UpdateDashboard

Awalan layanan	Tindakan
	iotsitewise: UpdateDataset
	iotsitewise: UpdateGateway
	iotsitewise: UpdateGatewayCapabilityConfiguration
	iotsitewise: UpdatePortal
	iotsitewise: UpdateProject

Awalan layanan	Tindakan
pembuat iottwinmaker	pembuat iottwinmaker: CancelMetadataTransferJob pembuat iottwinmaker: CreateComponentType pembuat iottwinmaker: CreateEntity pembuat iottwinmaker: CreateMetadataTransferJob pembuat iottwinmaker: CreateScene pembuat iottwinmaker: CreateSyncJob pembuat iottwinmaker: CreateWorkspace pembuat iottwinmaker: DeleteComponentType pembuat iottwinmaker: DeleteEntity pembuat iottwinmaker: DeleteScene pembuat iottwinmaker: DeleteSyncJob pembuat iottwinmaker: DeleteWorkspace pembuat iottwinmaker: ExecuteQuery pembuat iottwinmaker: GetMetadataTransferJob pembuat iottwinmaker: GetPricingPlan pembuat iottwinmaker: GetScene pembuat iottwinmaker: GetSyncJob pembuat iottwinmaker: ListComponentTypes pembuat iottwinmaker: ListComponents pembuat iottwinmaker: ListEntities pembuat iottwinmaker: ListMetadataTransferJobs

Awalan layanan	Tindakan
	pembuat iottwinmaker: ListProperties pembuat iottwinmaker: ListScenes pembuat iottwinmaker: ListSyncJobs pembuat iottwinmaker: ListSyncResources pembuat iottwinmaker: ListWorkspaces pembuat iottwinmaker: UpdateComponentType pembuat iottwinmaker: UpdateEntity pembuat iottwinmaker: UpdatePricingPlan pembuat iottwinmaker: UpdateScene pembuat iottwinmaker: UpdateWorkspace

Awalan layanan	Tindakan
iotwireless	iotwireless: AssociateAwsAccountWithPartnerAccount
	iotwireless: AssociateMulticastGroupWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithMulticastGroup
	iotwireless: AssociateWirelessDeviceWithThing
	iotwireless: AssociateWirelessGatewayWithCertificate
	iotwireless: AssociateWirelessGatewayWithThing
	iotwireless: CancelMulticastGroupSession
	iotwireless: CreateDestination
	iotwireless: CreateDeviceProfile
	iotwireless: CreateFuotaTask
	iotwireless: CreateMulticastGroup
	iotwireless: CreateNetworkAnalyzerConfiguration
	iotwireless: CreateServiceProfile
	iotwireless: CreateWirelessDevice
	iotwireless: CreateWirelessGateway
	iotwireless: CreateWirelessGatewayTask
	iotwireless: CreateWirelessGatewayTaskDefinition
	iotwireless: DeleteDestination
	iotwireless: DeleteDeviceProfile
	iotwireless: DeleteFuotaTask

Awalan layanan	Tindakan
	iotwireless: DeleteMulticastGroup
	iotwireless: DeleteNetworkAnalyzerConfiguration
	iotwireless: DeleteQueuedMessages
	iotwireless: DeleteServiceProfile
	iotwireless: DeleteWirelessDevice
	iotwireless: DeleteWirelessDeviceImportTask
	iotwireless: DeleteWirelessGateway
	iotwireless: DeleteWirelessGatewayTask
	iotwireless: DeleteWirelessGatewayTaskDefinition
	iotwireless: DeregisterWirelessDevice
	iotwireless: DisassociateAwsAccountFromPartnerAccount
	iotwireless: DisassociateMulticastGroupFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromMulticastGroup
	iotwireless: DisassociateWirelessDeviceFromThing
	iotwireless: DisassociateWirelessGatewayFromCertificate
	iotwireless: DisassociateWirelessGatewayFromThing
	iotwireless: GetDestination
	iotwireless: GetDeviceProfile
	iotwireless: GetEventConfigurationByResourceTypes
	iotwireless: GetFuotaTask

Awalan layanan	Tindakan
	iotwireless: GetLogLevelsByResourceTypes
	iotwireless: GetMetricConfiguration
	iotwireless: GetMetrics
	iotwireless: GetMulticastGroup
	iotwireless: GetMulticastGroupSession
	iotwireless: GetNetworkAnalyzerConfiguration
	iotwireless: GetPartnerAccount
	iotwireless: GetPosition
	iotwireless: GetPositionConfiguration
	iotwireless: GetPositionEstimate
	iotwireless: GetResourceEventConfiguration
	iotwireless: GetResourceLogLevel
	iotwireless: GetResourcePosition
	iotwireless: GetServiceEndpoint
	iotwireless: GetServiceProfile
	iotwireless: GetWirelessDevice
	iotwireless: GetWirelessDeviceImportTask
	iotwireless: GetWirelessDeviceStatistics
	iotwireless: GetWirelessGateway
	iotwireless: GetWirelessGatewayCertificate
	iotwireless: GetWirelessGatewayFirmwareInformation

Awalan layanan	Tindakan
	iotwireless: GetWirelessGatewayStatistics
	iotwireless: GetWirelessGatewayTask
	iotwireless: GetWirelessGatewayTaskDefinition
	iotwireless: ListDestinations
	iotwireless: ListDeviceProfiles
	iotwireless: ListDevicesForWirelessDeviceImportTask
	iotwireless: ListEventConfigurations
	iotwireless: ListFuotaTasks
	iotwireless: ListMulticastGroups
	iotwireless: ListMulticastGroupsByFuotaTask
	iotwireless: ListNetworkAnalyzerConfigurations
	iotwireless: ListPartnerAccounts
	iotwireless: ListPositionConfigurations
	iotwireless: ListQueuedMessages
	iotwireless: ListServiceProfiles
	iotwireless: ListWirelessDeviceImportTasks
	iotwireless: ListWirelessDevices
	iotwireless: ListWirelessGatewayTaskDefinitions
	iotwireless: ListWirelessGateways
	iotwireless: PutPositionConfiguration
	iotwireless: PutResourceLogLevel

Awalan layanan	Tindakan
	iotwireless: ResetAllResourceLogLevels
	iotwireless: ResetResourceLogLevel
	iotwireless: SendDataToMulticastGroup
	iotwireless: SendDataToWirelessDevice
	iotwireless: StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless: StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless: StartFuotaTask
	iotwireless: StartMulticastGroupSession
	iotwireless: StartNetworkAnalyzerStream
	iotwireless: StartSingleWirelessDeviceImportTask
	iotwireless: StartWirelessDeviceImportTask
	iotwireless: TestWirelessDevice
	iotwireless: UpdateDestination
	iotwireless: UpdateEventConfigurationByResourceTypes
	iotwireless: UpdateFuotaTask
	iotwireless: UpdateLogLevelsByResourceTypes
	iotwireless: UpdateMetricConfiguration
	iotwireless: UpdateMulticastGroup
	iotwireless: UpdateNetworkAnalyzerConfiguration
	iotwireless: UpdatePartnerAccount

Awalan layanan	Tindakan
	iotwireless: UpdatePosition
	iotwireless: UpdateResourceEventConfiguration
	iotwireless: UpdateResourcePosition
	iotwireless: UpdateWirelessDevice
	iotwireless: UpdateWirelessDeviceImportTask
	iotwireless: UpdateWirelessGateway

Awalan layanan	Tindakan
ivs	ivs: BatchGetChannel ivs: BatchGetStreamKey ivs: BatchStartViewerSessionRevocation ivs: CreateAdConfiguration ivs: CreateChannel ivs: CreateEncoderConfiguration ivs: CreateIngestConfiguration ivs: CreateParticipantToken ivs: CreatePlaybackRestrictionPolicy ivs: CreateRecordingConfiguration ivs: CreateStorageConfiguration ivs: CreateStreamKey ivs: DeleteAdConfiguration ivs: DeleteChannel ivs: DeleteEncoderConfiguration ivs: DeleteIngestConfiguration ivs: DeletePlaybackKeyPair ivs: DeletePlaybackRestrictionPolicy ivs: DeletePublicKey ivs: DeleteRecordingConfiguration ivs: DeleteStorageConfiguration

Awalan layanan	Tindakan
	ivs: DeleteStreamKey ivs: DisconnectParticipant ivs: GetAdConfiguration ivs: GetChannel ivs: GetComposition ivs: GetEncoderConfiguration ivs: GetIngestConfiguration ivs: GetParticipant ivs: GetPlaybackKeyPair ivs: GetPlaybackRestrictionPolicy ivs: GetPublicKey ivs: GetRecordingConfiguration ivs: GetStorageConfiguration ivs: GetStream ivs: GetStreamKey ivs: GetStreamSession ivs: ImportPlaybackKeyPair ivs: ImportPublicKey ivs: InsertAdBreak ivs: ListAdConfigurations ivs: ListChannels

Awalan layanan	Tindakan
	ivs: ListCompositions ivs: ListEncoderConfigurations ivs: ListIngestConfigurations ivs: ListParticipantEvents ivs: ListParticipantReplicas ivs: ListParticipants ivs: ListPlaybackKeyPairs ivs: ListPlaybackRestrictionPolicies ivs: ListPublicKeys ivs: ListRecordingConfigurations ivs: ListStorageConfigurations ivs: ListStreamKeys ivs: ListStreamSessions ivs: ListStreams ivs: PutMetadata ivs: StartComposition ivs: StartViewerSessionRevocation ivs: StopComposition ivs: StopStream ivs: UpdateAdConfiguration ivs: UpdateChannel

Awalan layanan	Tindakan
	ivs: UpdateIngestConfiguration ivs: UpdatePlaybackRestrictionPolicy
ivschat	ivschat: CreateChatToken ivschat: CreateLoggingConfiguration ivschat: CreateRoom ivschat: DeleteLoggingConfiguration ivschat: DeleteMessage ivschat: DeleteRoom ivschat: DisconnectUser ivschat: GetLoggingConfiguration ivschat: GetRoom ivschat: ListLoggingConfigurations ivschat: ListRooms ivschat: SendEvent ivschat: UpdateLoggingConfiguration ivschat: UpdateRoom

Awalan layanan	Tindakan
kafka	kafka: BatchAssociateScramSecret kafka: BatchDisassociateScramSecret kafka: CreateCluster kafka: CreateCluster V2 kafka: CreateConfiguration kafka: CreateReplicator kafka: CreateVpcConnection kafka: DeleteCluster kafka: DeleteClusterPolicy kafka: DeleteConfiguration kafka: DeleteReplicator kafka: DeleteVpcConnection kafka: DescribeCluster kafka: DescribeClusterOperation kafka: DescribeClusterOperationV2 kafka: DescribeCluster V2 kafka: DescribeConfiguration kafka: DescribeConfigurationRevision kafka: DescribeVpcConnection kafka: GetBootstrapBrokers kafka: GetClusterPolicy

Awalan layanan	Tindakan
	kafka: GetCompatibleKafkaVersions kafka: ListClientVpcConnections kafka: ListClusterOperations kafka: ListClusterOperationsV2 kafka: ListClusters kafka: ListClusters V2 kafka: ListConfigurationRevisions kafka: ListConfigurations kafka: ListKafkaVersions kafka: ListNodes kafka: ListReplicators kafka: ListScramSecrets kafka: ListVpcConnections kafka: PutClusterPolicy kafka: RebootBroker kafka: RejectClientVpcConnection kafka: UpdateBrokerCount kafka: UpdateBrokerStorage kafka: UpdateBrokerType kafka: UpdateClusterConfiguration kafka: UpdateClusterKafkaVersion

Awalan layanan	Tindakan
	kafka: UpdateConfiguration kafka: UpdateConnectivity kafka: UpdateMonitoring kafka: UpdateRebalancing kafka: UpdateReplicationInfo kafka: UpdateSecurity kafka: UpdateStorage
kafkaconnect	kafkaconnect: CreateConnector kafkaconnect: CreateCustomPlugin kafkaconnect: CreateWorkerConfiguration kafkaconnect: DeleteConnector kafkaconnect: DeleteCustomPlugin kafkaconnect: DeleteWorkerConfiguration kafkaconnect: DescribeConnector kafkaconnect: DescribeCustomPlugin kafkaconnect: DescribeWorkerConfiguration kafkaconnect: ListConnectorOperations kafkaconnect: ListConnectors kafkaconnect: ListCustomPlugins kafkaconnect: ListWorkerConfigurations kafkaconnect: UpdateConnector

Awalan layanan	Tindakan
kendra	kendra: AssociateEntitiesToExperience kendra: AssociatePersonasToEntities kendra: BatchDeleteDocument kendra: BatchDeleteFeaturedResultsSet kendra: BatchGetDocumentStatus kendra: BatchPutDocument kendra: ClearQuerySuggestions kendra: CreateAccessControlConfiguration kendra: CreateDataSource kendra: CreateExperience kendra: CreateFaq kendra: CreateFeaturedResultsSet kendra: CreateIndex kendra: CreateQuerySuggestionsBlockList kendra: CreateThesaurus kendra: DeleteDataSource kendra: DeleteExperience kendra: DeleteFaq kendra: DeleteIndex kendra: DeletePrincipalMapping kendra: DeleteQuerySuggestionsBlockList

Awalan layanan	Tindakan
	kendra: DeleteThesaurus kendra: DescribeAccessControlConfiguration kendra: DescribeDataSource kendra: DescribeExperience kendra: DescribeFaq kendra: DescribeFeaturedResultsSet kendra: DescribeIndex kendra: DescribePrincipalMapping kendra: DescribeQuerySuggestionsBlockList kendra: DescribeQuerySuggestionsConfig kendra: DescribeThesaurus kendra: DisassociateEntitiesFromExperience kendra: DisassociatePersonasFromEntities kendra: GetQuerySuggestions kendra: GetSnapshots kendra: ListAccessControlConfigurations kendra: ListDataSourceSyncJobs kendra: ListDataSources kendra: ListEntityPersonas kendra: ListExperienceEntities kendra: ListExperiences

Awalan layanan	Tindakan
	kendra: ListFaqs kendra: ListFeaturedResultsSets kendra: ListGroupsOlderThanOrderingId kendra: ListIndices kendra: ListQuerySuggestionsBlockLists kendra: ListThesauri kendra: PutPrincipalMapping Kendra:Pertanyaan Kendra:ambil kendra: StartDataSourceSyncJob kendra: StopDataSourceSyncJob kendra: SubmitFeedback kendra: UpdateDataSource kendra: UpdateExperience kendra: UpdateFeaturedResultsSet kendra: UpdateIndex kendra: UpdateQuerySuggestionsBlockList kendra: UpdateQuerySuggestionsConfig kendra: UpdateThesaurus

Awalan layanan	Tindakan
kinesis	kinesis: CreateStream
	kinesis: DecreaseStreamRetentionPeriod
	kinesis: DeleteStream
	kinesis: DeregisterStreamConsumer
	kinesis: DescribeAccountSettings
	kinesis: DescribeLimits
	kinesis: DescribeStream
	kinesis: DescribeStreamConsumer
	kinesis: DescribeStreamSummary
	kinesis: DisableEnhancedMonitoring
	kinesis: EnableEnhancedMonitoring
	kinesis: IncreaseStreamRetentionPeriod
	kinesis: ListShards
	kinesis: ListStreamConsumers
	kinesis: ListStreams
	kinesis: MergeShards
	kinesis: RegisterStreamConsumer
	kinesis: SplitShard
	kinesis: StartStreamEncryption
	kinesis: StopStreamEncryption
	kinesis: UpdateAccountSettings

Awalan layanan	Tindakan
	kinesis: UpdateShardCount kinesis: UpdateStreamMode

Awalan layanan	Tindakan
kinesisanalitik	kinesisanalitik: AddApplicationCloudWatchLoggingOption kinesisanalitik: AddApplicationInput kinesisanalitik: AddApplicationInputProcessingConfiguration kinesisanalitik: AddApplicationOutput kinesisanalitik: AddApplicationReferenceDataSource kinesisanalitik: AddApplicationVpcConfiguration kinesisanalitik: CreateApplication kinesisanalitik: CreateApplicationPresignedUrl kinesisanalitik: CreateApplicationSnapshot kinesisanalitik: DeleteApplication kinesisanalitik: DeleteApplicationCloudWatchLoggingOption kinesisanalitik: DeleteApplicationInputProcessingConfiguration kinesisanalitik: DeleteApplicationOutput kinesisanalitik: DeleteApplicationReferenceDataSource kinesisanalitik: DeleteApplicationSnapshot kinesisanalitik: DeleteApplicationVpcConfiguration kinesisanalitik: DescribeApplication kinesisanalitik: DescribeApplicationOperation kinesisanalitik: DescribeApplicationSnapshot kinesisanalitik: DescribeApplicationVersion kinesisanalitik: DiscoverInputSchema

Awalan layanan	Tindakan
	kinesisanalitik: ListApplicationOperations kinesisanalitik: ListApplicationSnapshots kinesisanalitik: ListApplicationVersions kinesisanalitik: ListApplications kinesisanalitik: RollbackApplication kinesisanalitik: StartApplication kinesisanalitik: StopApplication kinesisanalitik: UpdateApplication kinesisanalitik: UpdateApplicationMaintenanceConfiguration

Awalan layanan	Tindakan
km	km: CancelKeyDeletion
	km: ConnectCustomKeyStore
	km: CreateAlias
	km: CreateCustomKeyStore
	km: CreateGrant
	km: CreateKey
	kms:Decrypt
	km: DeleteAlias
	km: DeleteCustomKeyStore
	km: DeleteImportedKeyMaterial
	km: DeriveSharedSecret
	km: DescribeCustomKeyStores
	km: DescribeKey
	km: DisableKey
	km: DisableKeyRotation
	km: DisconnectCustomKeyStore
	km: EnableKey
	km: EnableKeyRotation
	kms:Encrypt
	km: GenerateDataKey
	km: GenerateDataKeyPair

Awalan layanan	Tindakan
	km: GenerateDataKeyPairWithoutPlaintext
	km: GenerateDataKeyWithoutPlaintext
	km: GenerateMac
	km: GenerateRandom
	km: GetKeyLastUsage
	km: GetKeyPolicy
	km: GetKeyRotationStatus
	km: GetParametersForImport
	km: GetPublicKey
	km: ImportKeyMaterial
	km: ListAliases
	km: ListGrants
	km: ListKeyPolicies
	km: ListKeyRotations
	km: ListKeys
	km: ListRetirableGrants
	km: ReplicateKey
	km: RetireGrant
	km: RevokeGrant
	km: RotateKeyOnDemand
	km: ScheduleKeyDeletion

Awalan layanan	Tindakan
	KMS: Tanda
	km: UpdateAlias
	km: UpdateCustomKeyStore
	km: UpdateKeyDescription
	km: UpdatePrimaryRegion
	KMS: Verifikasi
	km: VerifyMac

Awalan layanan	Tindakan
lambda	lambda: AddLayerVersionPermission lambda: AddPermission lambda: CreateAlias lambda: CreateCodeSigningConfig lambda: CreateEventSourceMapping lambda: CreateFunction lambda: CreateFunctionUrlConfig lambda: DeleteAlias lambda: DeleteCapacityProvider lambda: DeleteCodeSigningConfig lambda: DeleteEventSourceMapping lambda: DeleteFunction lambda: DeleteFunctionCodeSigningConfig lambda: DeleteFunctionConcurrency lambda: DeleteFunctionEventInvokeConfig lambda: DeleteFunctionUrlConfig lambda: DeleteLayerVersion lambda: DeleteProvisionedConcurrencyConfig lambda: GetAccountSettings lambda: GetAlias lambda: GetCapacityProvider

Awalan layanan	Tindakan
	lambda: GetCodeSigningConfig lambda: GetEventSourceMapping lambda: GetFunction lambda: GetFunctionCodeSigningConfig lambda: GetFunctionConcurrency lambda: GetFunctionConfiguration lambda: GetFunctionEventInvokeConfig lambda: GetFunctionRecursionConfig lambda: GetFunctionScalingConfig lambda: GetFunctionUrlConfig lambda: GetLayerVersion lambda: GetLayerVersionPolicy lambda: GetPolicy lambda: GetProvisionedConcurrencyConfig lambda: GetRuntimeManagementConfig lambda: ListAliases lambda: ListCapacityProviders lambda: ListCodeSigningConfigs lambda: ListDurableExecutionsByFunction lambda: ListEventSourceMappings lambda: ListFunctionEventInvokeConfigs

Awalan layanan	Tindakan
	lambda: ListFunctionUrlConfigs lambda: ListFunctions lambda: ListFunctionsByCodeSigningConfig lambda: ListLayerVersions lambda: ListLayers lambda: ListProvisionedConcurrencyConfigs lambda: ListVersionsByFunction lambda: PublishLayerVersion lambda: PublishVersion lambda: PutFunctionCodeSigningConfig lambda: PutFunctionConcurrency lambda: PutFunctionEventInvokeConfig lambda: PutFunctionRecursionConfig lambda: PutFunctionScalingConfig lambda: PutProvisionedConcurrencyConfig lambda: PutRuntimeManagementConfig lambda: RemoveLayerVersionPermission lambda: RemovePermission lambda: UpdateAlias lambda: UpdateCapacityProvider lambda: UpdateCodeSigningConfig

Awalan layanan	Tindakan
	lambda: UpdateEventSourceMapping lambda: UpdateFunctionCode lambda: UpdateFunctionConfiguration lambda: UpdateFunctionEventInvokeConfig lambda: UpdateFunctionUrlConfig

Awalan layanan	Tindakan
lex	lex: BatchCreateCustomVocabularyItem
	lex: BatchDeleteCustomVocabularyItem
	lex: BatchUpdateCustomVocabularyItem
	lex: BuildBotLocale
	lex: CreateBotAlias
	lex: CreateBotReplica
	lex: CreateBotVersion
	lex: CreateExport
	lex: CreateIntentVersion
	lex: CreateResourcePolicy
	lex: CreateSlotTypeVersion
	lex: CreateTestSetDiscrepancyReport
	lex: CreateUploadUrl
	lex: DeleteBot
	lex: DeleteBotAnalyzerRecommendation
	lex: DeleteBotChannelAssociation
	lex: DeleteBotReplica
	lex: DeleteExport
	lex: DeleteImport
	lex: DeleteIntentVersion
	lex: DeleteResourcePolicy

Awalan layanan	Tindakan
	lex: DeleteSlotTypeVersion
	lex: DeleteTestSet
	lex: DeleteUtterances
	lex: DescribeBotAlias
	lex: DescribeBotAnalyzerRecommendation
	lex: DescribeBotRecommendation
	lex: DescribeBotReplica
	lex: DescribeBotResourceGeneration
	lex: DescribeBotVersion
	lex: DescribeCustomVocabularyMetadata
	lex: DescribeExport
	lex: DescribeImport
	lex: DescribeResourcePolicy
	lex: DescribeTestExecution
	lex: DescribeTestSet
	lex: DescribeTestSetDiscrepancyReport
	lex: DescribeTestSetGeneration
	lex: GenerateBotElement
	lex: GetBot
	lex: GetBotAlias
	lex: GetBotAliases

Awalan layanan	Tindakan
	lex: GetBotChannelAssociation
	lex: GetBotChannelAssociations
	lex: GetBotVersions
	lex: GetBots
	lex: GetBuiltinIntent
	lex: GetBuiltinIntents
	lex: GetBuiltinSlotTypes
	lex: GetExport
	lex: GetImport
	lex: GetIntent
	lex: GetIntentVersions
	lex: GetIntents
	lex: GetMigration
	lex: GetMigrations
	lex: GetSlotType
	lex: GetSlotTypeVersions
	lex: GetSlotTypes
	lex: GetTestExecutionArtifactsUrl
	lex: GetUtterancesView
	lex: ListBotAliasReplicas
	lex: ListBotAliases

Awalan layanan	Tindakan
	lex: ListBotRecommendations
	lex: ListBotReplicas
	lex: ListBotResourceGenerations
	lex: ListBotVersionReplicas
	lex: ListBotVersions
	lex: ListBots
	lex: ListBuiltInIntents
	lex: ListBuiltInSlotTypes
	lex: ListCustomVocabularyItems
	lex: ListExports
	lex: ListImports
	lex: ListIntentMetrics
	lex: ListIntentPaths
	lex: ListRecommendedIntents
	lex: ListSessionAnalyticsData
	lex: ListSessionMetrics
	lex: ListTestExecutionResultItems
	lex: ListTestExecutions
	lex: ListTestSets
	lex: PutBot
	lex: PutBotAlias

Awalan layanan	Tindakan
	lex: PutIntent
	lex: PutSlotType
	lex: SearchAssociatedTranscripts
	lex: StartBotAnalyzer
	lex: StartBotRecommendation
	lex: StartImport
	lex: StartMigration
	lex: StartTestExecution
	lex: StartTestSetGeneration
	lex: StopBotAnalyzer
	lex: StopBotRecommendation
	lex: UpdateBotAlias
	lex: UpdateBotRecommendation
	lex: UpdateExport
	lex: UpdateResourcePolicy

Awalan layanan	Tindakan
lisensi-manajer-linux-langganan	lisensi-manajer-linux-langganan: DeregisterSubscriptionProvider lisensi-manajer-linux-langganan: GetRegisteredSubscriptionProvider lisensi-manajer-linux-langganan: GetServiceSettings lisensi-manajer-linux-langganan: ListLinuxSubscriptionInstances lisensi-manajer-linux-langganan: ListLinuxSubscriptions lisensi-manajer-linux-langganan: ListRegisteredSubscriptionProviders lisensi-manajer-linux-langganan: RegisterSubscriptionProvider lisensi-manajer-linux-langganan: UpdateServiceSettings

Awalan layanan	Tindakan
lightsail	lightsail: AllocateStaticIp
	lightsail: AttachCertificateToDistribution
	lightsail: AttachDisk
	lightsail: AttachInstancesToLoadBalancer
	lightsail: AttachLoadBalancerTlsCertificate
	lightsail: AttachStaticIp
	lightsail: CloseInstancePublicPorts
	lightsail: CopySnapshot
	lightsail: CreateBucket
	lightsail: CreateBucketAccessKey
	lightsail: CreateCertificate
	lightsail: CreateCloudFormationStack
	lightsail: CreateContactMethod
	lightsail: CreateContainerService
	lightsail: CreateContainerServiceDeployment
	lightsail: CreateContainerServiceRegistryLogin
	lightsail: CreateDisk
	lightsail: CreateDiskFromSnapshot
	lightsail: CreateDiskSnapshot
	lightsail: CreateDistribution
	lightsail: CreateDomain

Awalan layanan	Tindakan
	lightsail: CreateGUISessionAccessDetails
	lightsail: CreateInstanceSnapshot
	lightsail: CreateInstances
	lightsail: CreateInstancesFromSnapshot
	lightsail: CreateKeyPair
	lightsail: CreateLoadBalancer
	lightsail: CreateLoadBalancerTlsCertificate
	lightsail: CreateRelationalDatabase
	lightsail: CreateRelationalDatabaseFromSnapshot
	lightsail: CreateRelationalDatabaseSnapshot
	lightsail: DeleteAlarm
	lightsail: DeleteAutoSnapshot
	lightsail: DeleteBucket
	lightsail: DeleteBucketAccessKey
	lightsail: DeleteCertificate
	lightsail: DeleteContactMethod
	lightsail: DeleteContainerImage
	lightsail: DeleteContainerService
	lightsail: DeleteDisk
	lightsail: DeleteDiskSnapshot
	lightsail: DeleteDistribution

Awalan layanan	Tindakan
	lightsail: DeleteDomain
	lightsail: DeleteDomainEntry
	lightsail: DeleteInstance
	lightsail: DeleteInstanceSnapshot
	lightsail: DeleteKeyPair
	lightsail: DeleteKnownHostKeys
	lightsail: DeleteLoadBalancer
	lightsail: DeleteLoadBalancerTlsCertificate
	lightsail: DeleteRelationalDatabase
	lightsail: DeleteRelationalDatabaseSnapshot
	lightsail: DetachCertificateFromDistribution
	lightsail: DetachDisk
	lightsail: DetachInstancesFromLoadBalancer
	lightsail: DetachStaticIp
	lightsail: DisableAddOn
	lightsail: DownloadDefaultKeyPair
	lightsail: EnableAddOn
	lightsail: ExportSnapshot
	lightsail: GetActiveNames
	lightsail: GetAlarms
	lightsail: GetAutoSnapshots

Awalan layanan	Tindakan
	lightsail: GetBlueprints
	lightsail: GetBucketAccessKeys
	lightsail: GetBucketBundles
	lightsail: GetBucketMetricData
	lightsail: GetBuckets
	lightsail: GetBundles
	lightsail: GetCertificates
	lightsail: GetCloudFormationStackRecords
	lightsail: GetContactMethods
	lightsail: GetContainerAPIMetadata
	lightsail: GetContainerImages
	lightsail: GetContainerLog
	lightsail: GetContainerServiceDeployments
	lightsail: GetContainerServiceMetricData
	lightsail: GetContainerServicePowers
	lightsail: GetContainerServices
	lightsail: GetCostEstimate
	lightsail: GetDisk
	lightsail: GetDiskSnapshot
	lightsail: GetDiskSnapshots
	lightsail: GetDisks

Awalan layanan	Tindakan
	lightsail: GetDistributionBundles
	lightsail: GetDistributionLatestCacheReset
	lightsail: GetDistributionMetricData
	lightsail: GetDistributions
	lightsail: GetDomain
	lightsail: GetExportSnapshotRecords
	lightsail: GetInstance
	lightsail: GetInstanceMetricData
	lightsail: GetInstancePortStates
	lightsail: GetInstanceSnapshot
	lightsail: GetInstanceSnapshots
	lightsail: GetInstanceState
	lightsail: GetInstances
	lightsail: GetKeyPair
	lightsail: GetKeyPairs
	lightsail: GetLoadBalancer
	lightsail: GetLoadBalancerMetricData
	lightsail: GetLoadBalancerTlsCertificates
	lightsail: GetLoadBalancerTlsPolicies
	lightsail: GetLoadBalancers
	lightsail: GetOperation

Awalan layanan	Tindakan
	lightsail: GetOperations
	lightsail: GetOperationsForResource
	lightsail: GetRegions
	lightsail: GetRelationalDatabase
	lightsail: GetRelationalDatabaseBlueprints
	lightsail: GetRelationalDatabaseBundles
	lightsail: GetRelationalDatabaseEvents
	lightsail: GetRelationalDatabaseLogEvents
	lightsail: GetRelationalDatabaseLogStreams
	lightsail: GetRelationalDatabaseMasterUserPassword
	lightsail: GetRelationalDatabaseMetricData
	lightsail: GetRelationalDatabaseParameters
	lightsail: GetRelationalDatabaseSnapshot
	lightsail: GetRelationalDatabaseSnapshots
	lightsail: GetRelationalDatabases
	lightsail: GetSetupHistory
	lightsail: GetStaticIp
	lightsail: GetStaticIps
	lightsail: ImportKeyPair
	lightsail: IsVpcPeered
	lightsail: OpenInstancePublicPorts

Awalan layanan	Tindakan
	lightsail: PeerVpc
	lightsail: PutAlarm
	lightsail: PutInstancePublicPorts
	lightsail: RebootInstance
	lightsail: RebootRelationalDatabase
	lightsail: RegisterContainerImage
	lightsail: ReleaseStaticIp
	lightsail: ResetDistributionCache
	lightsail: SendContactMethodVerification
	lightsail: SetIpAddressType
	lightsail: SetResourceAccessForBucket
	lightsail: SetupInstanceHttps
	LightSail: startGuiSession
	lightsail: StartInstance
	lightsail: StartRelationalDatabase
	LightSail: StopGuiSession
	lightsail: StopInstance
	lightsail: StopRelationalDatabase
	lightsail: TestAlarm
	lightsail: UnpeerVpc
	lightsail: UpdateBucket

Awalan layanan	Tindakan
	lightsail: UpdateBucketBundle
	lightsail: UpdateContainerService
	lightsail: UpdateDistribution
	lightsail: UpdateDistributionBundle
	lightsail: UpdateDomainEntry
	lightsail: UpdateInstanceMetadataOptions
	lightsail: UpdateLoadBalancerAttribute
	lightsail: UpdateRelationalDatabase
	lightsail: UpdateRelationalDatabaseParameters

Awalan layanan	Tindakan
log	log: AssociateKmsKey log: AssociateSourceToS3TableIntegration log: CancelExportTask log: CancelImportTask log: CreateDelivery log: CreateExportTask log: CreateLogAnomalyDetector log: CreateLogGroup log: CreateLogStream log: CreateLookupTable log: DeleteDataProtectionPolicy log: DeleteDelivery log: DeleteDeliveryDestination log: DeleteDeliveryDestinationPolicy log: DeleteDeliverySource log: DeleteDestination log: DeleteIndexPolicy log: DeleteIntegration log: DeleteLogAnomalyDetector log: DeleteLogGroup log: DeleteLogStream

Awalan layanan	Tindakan
	log: DeleteLookupTable
	log: DeleteMetricFilter
	log: DeleteQueryDefinition
	log: DeleteResourcePolicy
	log: DeleteRetentionPolicy
	log: DeleteScheduledQuery
	log: DeleteSubscriptionFilter
	log: DeleteSyslogConfiguration
	log: DeleteTransformer
	log: DescribeAccountPolicies
	log: DescribeConfigurationTemplates
	log: DescribeDeliveries
	log: DescribeDeliveryDestinations
	log: DescribeDeliverySources
	log: DescribeDestinations
	log: DescribeExportTasks
	log: DescribeFieldIndexes
	log: DescribeImportTaskBatches
	log: DescribeImportTasks
	log: DescribeIndexPolicies
	log: DescribeLogGroups

Awalan layanan	Tindakan
	log: DescribeLogStreams log: DescribeLookupTables log: DescribeMetricFilters log: DescribeQueries log: DescribeQueryDefinitions log: DescribeResourcePolicies log: DescribeSubscriptionFilters log: DisassociateKmsKey log: DisassociateSourceFromS3TableIntegration log: GetDataProtectionPolicy log: GetDelivery log: GetDeliveryDestination log: GetDeliveryDestinationPolicy log: GetDeliverySource log: GetIntegration log: GetLogAnomalyDetector log: GetLogFields log: GetLogGroupFields log: GetLogRecord log: GetLookupTable log: GetQueryResults

Awalan layanan	Tindakan
	log: GetScheduledQuery log: GetScheduledQueryHistory log: GetStorageTierPolicy log: GetTransformer log: ListAnomalies log: ListIntegrations log: ListLogAnomalyDetectors log: ListLogGroupsForQuery log: ListScheduledQueries log: ListSourcesForS3TableIntegration log: ListSyslogConfigurations log: PutBearerTokenAuthentication log: PutDataProtectionPolicy log: PutDeliveryDestination log: PutDeliveryDestinationPolicy log: PutDeliverySource log: PutDestination log: PutDestinationPolicy log: PutIndexPolicy log: PutIntegration log: PutLogGroupDeletionProtection

Awalan layanan	Tindakan
	log: PutMetricFilter log: PutQueryDefinition log: PutResourcePolicy log: PutRetentionPolicy log: PutStorageTierPolicy log: PutSubscriptionFilter log: PutSyslogConfiguration log: PutTransformer log: StartLiveTail log: StartQuery log: StopQuery log: TestMetricFilter log: TestTransformer log: UpdateAnomaly log: UpdateDeliveryConfiguration log: UpdateLogAnomalyDetector log: UpdateLookupTable

Awalan layanan	Tindakan
peralatan pencarian	Lookoutequipment: CreateDataset
	Lookoutequipment: CreateInferenceScheduler
	Lookoutequipment: CreateLabel
	Lookoutequipment: CreateLabelGroup
	Lookoutequipment: CreateModel
	Lookoutequipment: DeleteDataset
	Lookoutequipment: DeleteInferenceScheduler
	Lookoutequipment: DeleteLabel
	Lookoutequipment: DeleteLabelGroup
	Lookoutequipment: DeleteModel
	Lookoutequipment: DeleteResourcePolicy
	Lookoutequipment: DeleteRetrainingScheduler
	Lookoutequipment: DescribeDataIngestionJob
	Lookoutequipment: DescribeDataset
	Lookoutequipment: DescribeInferenceScheduler
	Lookoutequipment: DescribeLabelGroup
	Lookoutequipment: DescribeModel
	Lookoutequipment: DescribeModelVersion
	Lookoutequipment: DescribeResourcePolicy
	Lookoutequipment: DescribeRetrainingScheduler
	LookouteQuipment:Describelabel

Awalan layanan	Tindakan
	Lookoutequipment: ImportDataset
	Lookoutequipment: ImportModelVersion
	Lookoutequipment: ListDataIngestionJobs
	Lookoutequipment: ListDatasets
	Lookoutequipment: ListInferenceEvents
	Lookoutequipment: ListInferenceExecutions
	Lookoutequipment: ListInferenceSchedulers
	Lookoutequipment: ListLabelGroups
	Lookoutequipment: ListLabels
	Lookoutequipment: ListModelVersions
	Lookoutequipment: ListModels
	Lookoutequipment: ListRetrainingSchedulers
	Lookoutequipment: ListSensorStatistics
	Lookoutequipment: PutResourcePolicy
	Lookoutequipment: StartDataIngestionJob
	Lookoutequipment: StartInferenceScheduler
	Lookoutequipment: StartRetrainingScheduler
	Lookoutequipment: StopInferenceScheduler
	Lookoutequipment: StopRetrainingScheduler
	Lookoutequipment: UpdateActiveModelVersion
	Lookoutequipment: UpdateInferenceScheduler

Awalan layanan	Tindakan
	Lookoutequipment: UpdateLabelGroup Lookoutequipment: UpdateModel Lookoutequipment: UpdateRetrainingScheduler

Awalan layanan	Tindakan
lookoutmetrics	lookoutmetrics: ActivateAnomalyDetector lookoutmetrics: BackTestAnomalyDetector lookoutmetrics: CreateAlert lookoutmetrics: CreateAnomalyDetector lookoutmetrics: CreateMetricSet lookoutmetrics: DeactivateAnomalyDetector lookoutmetrics: DeleteAlert lookoutmetrics: DeleteAnomalyDetector lookoutmetrics: DescribeAlert lookoutmetrics: DescribeAnomalyDetectionExecutions lookoutmetrics: DescribeAnomalyDetector lookoutmetrics: DescribeMetricSet lookoutmetrics: DetectMetricSetConfig lookoutmetrics: GetAnomalyGroup lookoutmetrics: GetDataQualityMetrics lookoutmetrics: GetFeedback lookoutmetrics: GetSampleData lookoutmetrics: ListAlerts lookoutmetrics: ListAnomalyDetectors lookoutmetrics: ListAnomalyGroupRelatedMetrics lookoutmetrics: ListAnomalyGroupSummaries

Awalan layanan	Tindakan
	lookoutmetrics: ListAnomalyGroupTimeSeries lookoutmetrics: ListMetricSets lookoutmetrics: PutFeedback lookoutmetrics: UpdateAlert lookoutmetrics: UpdateAnomalyDetector lookoutmetrics: UpdateMetricSet

Awalan layanan	Tindakan
m2	m2: CancelBatchJobExecution
	m2: CreateApplication
	m2: CreateDataSetExportTask
	m2: CreateDataSetImportTask
	m2: CreateDeployment
	m2: CreateEnvironment
	m2: DeleteApplication
	m2: DeleteApplicationFromEnvironment
	m2: DeleteEnvironment
	m2: GetApplication
	m2: GetApplicationVersion
	m2: GetBatchJobExecution
	m2: GetDataSetDetails
	m2: GetDataSetExportTask
	m2: GetDataSetImportTask
	m2: GetDeployment
	m2: GetEnvironment
	m2: GetSignedBluinsightsUrl
	m2: ListApplicationVersions
	m2: ListApplications
	m2: ListBatchJobDefinitions

Awalan layanan	Tindakan
	m2: ListBatchJobExecutions
	m2: ListBatchJobRestartPoints
	m2: ListDataSetExportHistory
	m2: ListDataSetImportHistory
	m2: ListDataSets
	m2: ListDeployments
	m2: ListEngineVersions
	m2: ListEnvironments
	m2: StartApplication
	m2: StartBatchJob
	m2: StopApplication
	m2: UpdateApplication
	m2: UpdateEnvironment

Awalan layanan	Tindakan
terkelolablockchain	terkelolablockchain: CreateAccessor terkelolablockchain: CreateMember terkelolablockchain: CreateNetwork terkelolablockchain: CreateNode terkelolablockchain: CreateProposal terkelolablockchain: DeleteAccessor terkelolablockchain: DeleteMember terkelolablockchain: DeleteNode terkelolablockchain: GetAccessor terkelolablockchain: GetMember terkelolablockchain: GetNetwork terkelolablockchain: GetNode terkelolablockchain: GetProposal terkelolablockchain: InvokeRpcPolygonMainnet terkelolablockchain: InvokeRpcPolygonMumbaiTestnet terkelolablockchain: ListAccessors terkelolablockchain: ListInvitations terkelolablockchain: ListMembers terkelolablockchain: ListNetworks terkelolablockchain: ListNodes terkelolablockchain: ListProposalVotes

Awalan layanan	Tindakan
	terkelolablockchain: ListProposals terkelolablockchain: RejectInvitation terkelolablockchain: UpdateMember terkelolablockchain: UpdateNode terkelolablockchain: VoteOnProposal

Awalan layanan	Tindakan
mediaconnect	mediaconnect: AddBridgeOutputs
	mediaconnect: AddBridgeSources
	mediaconnect: AddFlowMediaStreams
	mediaconnect: AddFlowOutputs
	mediaconnect: AddFlowSources
	mediaconnect: AddFlowVpcInterfaces
	mediaconnect: CreateBridge
	mediaconnect: CreateFlow
	mediaconnect: CreateGateway
	mediaconnect: DeleteBridge
	mediaconnect: DeleteFlow
	mediaconnect: DeleteGateway
	mediaconnect: DeleteRouterInput
	mediaconnect: DeleteRouterNetworkInterface
	mediaconnect: DeleteRouterOutput
	mediaconnect: DeregisterGatewayInstance
	mediaconnect: DescribeBridge
	mediaconnect: DescribeFlow
	mediaconnect: DescribeFlowSourceMetadata
	mediaconnect: DescribeFlowSourceThumbnail
	mediaconnect: DescribeGateway

Awalan layanan	Tindakan
	mediaconnect: DescribeGatewayInstance
	mediaconnect: DescribeOffering
	mediaconnect: DescribeReservation
	mediaconnect: GetRouterInput
	mediaconnect: GetRouterInputSourceMetadata
	mediaconnect: GetRouterInputThumbnail
	mediaconnect: GetRouterNetworkInterface
	mediaconnect: GetRouterOutput
	mediaconnect: GrantFlowEntitlements
	mediaconnect: ListBridges
	mediaconnect: ListEntitlements
	mediaconnect: ListFlows
	mediaconnect: ListGatewayInstances
	mediaconnect: ListGateways
	mediaconnect: ListOfferings
	mediaconnect: ListReservations
	mediaconnect: ListRouterInputs
	mediaconnect: ListRouterNetworkInterfaces
	mediaconnect: ListRouterOutputs
	mediaconnect: PurchaseOffering
	mediaconnect: RemoveBridgeOutput

Awalan layanan	Tindakan
	mediaconnect: RemoveBridgeSource
	mediaconnect: RemoveFlowMediaStream
	mediaconnect: RemoveFlowOutput
	mediaconnect: RemoveFlowSource
	mediaconnect: RemoveFlowVpcInterface
	mediaconnect: RestartRouterInput
	mediaconnect: RestartRouterOutput
	mediaconnect: RevokeFlowEntitlement
	mediaconnect: StartFlow
	mediaconnect: StartRouterInput
	mediaconnect: StartRouterOutput
	mediaconnect: StopFlow
	mediaconnect: StopRouterInput
	mediaconnect: StopRouterOutput
	mediaconnect: TakeRouterInput
	mediaconnect: UpdateBridge
	mediaconnect: UpdateBridgeOutput
	mediaconnect: UpdateBridgeSource
	mediaconnect: UpdateBridgeState
	mediaconnect: UpdateFlow
	mediaconnect: UpdateFlowEntitlement

Awalan layanan	Tindakan
	mediaconnect: UpdateFlowMediaStream mediaconnect: UpdateGatewayInstance

Awalan layanan	Tindakan
mediaconvert	mediaconvert: AssociateCertificate mediaconvert: CancelJob mediaconvert: CreateJob mediaconvert: CreateJobTemplate mediaconvert: CreatePreset mediaconvert: CreateQueue mediaconvert: CreateResourceShare mediaconvert: DeleteJobTemplate mediaconvert: DeletePolicy mediaconvert: DeletePreset mediaconvert: DeleteQueue mediaconvert: DescribeEndpoints mediaconvert: DisassociateCertificate mediaconvert: GetJob mediaconvert: GetJobTemplate mediaconvert: GetPolicy mediaconvert: GetPreset mediaconvert: GetQueue mediaconvert: ListJobTemplates mediaconvert: ListJobs mediaconvert: ListPresets

Awalan layanan	Tindakan
	mediaconvert: ListQueues mediaconvert: ListVersions MediaConvert:Probe mediaconvert: PutPolicy mediaconvert: SearchJobs mediaconvert: UpdateJobTemplate mediaconvert: UpdatePreset mediaconvert: UpdateQueue

Awalan layanan	Tindakan
medialive	medialive: AcceptInputDeviceTransfer
	medialive: BatchDelete
	medialive: BatchStart
	medialive: BatchStop
	medialive: BatchUpdateSchedule
	medialive: CancelInputDeviceTransfer
	medialive: ClaimDevice
	medialive: CreateChannel
	medialive: CreateChannelPlacementGroup
	medialive: CreateCloudWatchAlarmTemplate
	medialive: CreateCloudWatchAlarmTemplateGroup
	medialive: CreateCluster
	medialive: CreateEventBridgeRuleTemplate
	medialive: CreateEventBridgeRuleTemplateGroup
	medialive: CreateInput
	medialive: CreateInputSecurityGroup
	medialive: CreateMultiplex
	medialive: CreateMultiplexProgram
	medialive: CreateNetwork
	medialive: CreateNode
	medialive: CreateNodeRegistrationScript

Awalan layanan	Tindakan
	medialive: CreatePartnerInput
	medialive: CreateSdiSource
	medialive: CreateSignalMap
	medialive: DeleteChannel
	medialive: DeleteChannelPlacementGroup
	medialive: DeleteCloudWatchAlarmTemplate
	medialive: DeleteCloudWatchAlarmTemplateGroup
	medialive: DeleteCluster
	medialive: DeleteEventBridgeRuleTemplate
	medialive: DeleteEventBridgeRuleTemplateGroup
	medialive: DeleteInput
	medialive: DeleteInputSecurityGroup
	medialive: DeleteMultiplex
	medialive: DeleteMultiplexProgram
	medialive: DeleteNetwork
	medialive: DeleteNode
	medialive: DeleteReservation
	medialive: DeleteSchedule
	medialive: DeleteSdiSource
	medialive: DeleteSignalMap
	medialive: DescribeAccountConfiguration

Awalan layanan	Tindakan
	medialive: DescribeChannel
	medialive: DescribeChannelPlacementGroup
	medialive: DescribeCluster
	medialive: DescribeInput
	medialive: DescribeInputDevice
	medialive: DescribeInputDeviceThumbnail
	medialive: DescribeInputSecurityGroup
	medialive: DescribeMultiplex
	medialive: DescribeMultiplexProgram
	medialive: DescribeNetwork
	medialive: DescribeNode
	medialive: DescribeOffering
	medialive: DescribeReservation
	medialive: DescribeSchedule
	medialive: DescribeSdiSource
	medialive: DescribeThumbnails
	medialive: GetCloudWatchAlarmTemplate
	medialive: GetCloudWatchAlarmTemplateGroup
	medialive: GetEventBridgeRuleTemplate
	medialive: GetEventBridgeRuleTemplateGroup
	medialive: GetSignalMap

Awalan layanan	Tindakan
	medialive: ListAlerts
	medialive: ListChannelPlacementGroups
	medialive: ListChannels
	medialive: ListCloudWatchAlarmTemplateGroups
	medialive: ListCloudWatchAlarmTemplates
	medialive: ListClusterAlerts
	medialive: ListClusters
	medialive: ListEventBridgeRuleTemplateGroups
	medialive: ListEventBridgeRuleTemplates
	medialive: ListInputDeviceTransfers
	medialive: ListInputDevices
	medialive: ListInputSecurityGroups
	medialive: ListInputs
	medialive: ListMultiplexAlerts
	medialive: ListMultiplexPrograms
	medialive: ListMultiplexes
	medialive: ListNetworks
	medialive: ListNodes
	medialive: ListOfferings
	medialive: ListReservations
	medialive: ListSdiSources

Awalan layanan	Tindakan
	medialive: ListSignalMaps
	medialive: ListVersions
	medialive: PurchaseOffering
	medialive: RebootInputDevice
	medialive: RejectInputDeviceTransfer
	medialive: RestartChannelPipelines
	medialive: StartChannel
	medialive: StartDeleteMonitorDeployment
	medialive: StartInputDevice
	medialive: StartInputDeviceMaintenanceWindow
	medialive: StartMonitorDeployment
	medialive: StartMultiplex
	medialive: StartUpdateSignalMap
	medialive: StopChannel
	medialive: StopInputDevice
	medialive: StopMultiplex
	medialive: TransferInputDevice
	medialive: UpdateAccountConfiguration
	medialive: UpdateChannel
	medialive: UpdateChannelClass
	medialive: UpdateChannelPlacementGroup

Awalan layanan	Tindakan
	medialive: UpdateCloudWatchAlarmTemplate
	medialive: UpdateCloudWatchAlarmTemplateGroup
	medialive: UpdateCluster
	medialive: UpdateEventBridgeRuleTemplate
	medialive: UpdateEventBridgeRuleTemplateGroup
	medialive: UpdateInput
	medialive: UpdateInputDevice
	medialive: UpdateInputSecurityGroup
	medialive: UpdateMultiplex
	medialive: UpdateMultiplexProgram
	medialive: UpdateNetwork
	medialive: UpdateNode
	medialive: UpdateNodeState
	medialive: UpdateReservation
	medialive: UpdateSdiSource

Awalan layanan	Tindakan
mediatailor	mediatailor: ConfigureLogsForPlaybackConfiguration mediatailor: CreateChannel mediatailor: CreateLiveSource mediatailor: CreatePrefetchSchedule mediatailor: CreateProgram mediatailor: CreateSourceLocation mediatailor: CreateVodSource mediatailor: DeleteChannel mediatailor: DeleteChannelPolicy mediatailor: DeleteLiveSource mediatailor: DeletePlaybackConfiguration mediatailor: DeletePrefetchSchedule mediatailor: DeleteProgram mediatailor: DeleteSourceLocation mediatailor: DeleteVodSource mediatailor: DescribeChannel mediatailor: DescribeLiveSource mediatailor: DescribeProgram mediatailor: DescribeSourceLocation mediatailor: DescribeVodSource mediatailor: GetChannelPolicy

Awalan layanan	Tindakan
	mediatailor: GetChannelSchedule mediatailor: GetPlaybackConfiguration mediatailor: GetPrefetchSchedule mediatailor: ListAlerts mediatailor: ListChannels mediatailor: ListLiveSources mediatailor: ListPlaybackConfigurations mediatailor: ListPrefetchSchedules mediatailor: ListSourceLocations mediatailor: ListVodSources mediatailor: PutChannelPolicy mediatailor: PutPlaybackConfiguration mediatailor: StartChannel mediatailor: StopChannel mediatailor: UpdateChannel mediatailor: UpdateLiveSource mediatailor: UpdateProgram mediatailor: UpdateSourceLocation mediatailor: UpdateVodSource

Awalan layanan	Tindakan
memorydb	memorydb: BatchUpdateCluster
	memorydb: CopySnapshot
	memorydb: CreateAcl
	memorydb: CreateCluster
	memorydb: CreateMultiRegionCluster
	memorydb: CreateParameterGroup
	memorydb: CreateSnapshot
	memorydb: CreateSubnetGroup
	memorydb: CreateUser
	memorydb: DeleteAcl
	memorydb: DeleteCluster
	memorydb: DeleteMultiRegionCluster
	memorydb: DeleteParameterGroup
	memorydb: DeleteSnapshot
	memorydb: DeleteSubnetGroup
	memorydb: DeleteUser
	memorydb: DescribeAcls
	memorydb: DescribeClusters
	memorydb: DescribeEngineVersions
	memorydb: DescribeEvents
	memorydb: DescribeMultiRegionClusters

Awalan layanan	Tindakan
	memorydb: DescribeMultiRegionParameterGroups
	memorydb: DescribeMultiRegionParameters
	memorydb: DescribeParameterGroups
	memorydb: DescribeParameters
	memorydb: DescribeReservedNodes
	memorydb: DescribeReservedNodesOfferings
	memorydb: DescribeServiceUpdates
	memorydb: DescribeSnapshots
	memorydb: DescribeSubnetGroups
	memorydb: DescribeUsers
	memorydb: FailoverShard
	memorydb: ListAllowedMultiRegionClusterUpdates
	memorydb: ListAllowedNodeTypeUpdates
	memorydb: PurchaseReservedNodesOffering
	memorydb: ResetParameterGroup
	memorydb: UpdateAcl
	memorydb: UpdateCluster
	memorydb: UpdateMultiRegionCluster
	memorydb: UpdateParameterGroup
	memorydb: UpdateSubnetGroup
	memorydb: UpdateUser

Awalan layanan	Tindakan
mgh	mgh: AssociateCreatedArtifact mgh: AssociateDiscoveredResource mgh: AssociateSourceResource mgh: CreateHomeRegionControl mgh: CreateProgressUpdateStream mgh: DeleteHomeRegionControl mgh: DeleteProgressUpdateStream mgh: DescribeApplicationState mgh: DescribeHomeRegionControls mgh: DescribeMigrationTask mgh: DisassociateCreatedArtifact mgh: DisassociateDiscoveredResource mgh: DisassociateSourceResource mgh: GetHomeRegion mgh: ImportMigrationTask mgh: ListApplicationStates mgh: ListCreatedArtifacts mgh: ListDiscoveredResources mgh: ListMigrationTaskUpdates mgh: ListMigrationTasks mgh: ListProgressUpdateStreams

Awalan layanan	Tindakan
	mgh: ListSourceResources mgh: NotifyApplicationState mgh: NotifyMigrationTaskState mgh: PutResourceAttributes

Awalan layanan	Tindakan
mgn	mgn: ArchiveApplication
	mgn: ArchiveWave
	mgn: AssociateApplications
	mgn: AssociateSourceServers
	mgn: ChangeServerLifeCycleState
	mgn: CreateApplication
	mgn: CreateConnector
	mgn: CreateLaunchConfigurationTemplate
	mgn: CreateNetworkMigrationDefinition
	mgn: CreateReplicationConfigurationTemplate
	mgn: CreateWave
	mgn: DeleteApplication
	mgn: DeleteConnector
	mgn: DeleteJob
	mgn: DeleteLaunchConfigurationTemplate
	mgn: DeleteNetworkMigrationDefinition
	mgn: DeleteReplicationConfigurationTemplate
	mgn: DeleteSourceServer
	mgn: DeleteVcenterClient
	mgn: DeleteWave
	mgn: DescribeJobLogItems

Awalan layanan	Tindakan
	mgn: DescribeJobs
	mgn: DescribeLaunchConfigurationTemplates
	mgn: DescribeReplicationConfigurationTemplates
	mgn: DescribeVcenterClients
	mgn: DisassociateApplications
	mgn: DisassociateSourceServers
	mgn: DisconnectFromService
	mgn: FinalizeCutover
	mgn: GetNetworkMigrationDefinition
	mgn: GetNetworkMigrationMapperSegmentConstruct
	mgn: GetReplicationConfiguration
	mgn: InitializeService
	mgn: ListConnectors
	mgn: ListExportErrors
	mgn: ListExports
	mgn: ListImportErrors
	mgn: ListImportFileEnrichments
	mgn: ListImports
	mgn: ListManagedAccounts
	mgn: ListNetworkMigrationAnalyses
	mgn: ListNetworkMigrationAnalysisResults

Awalan layanan	Tindakan
	mgn: ListNetworkMigrationCodeGenerationSegments
	mgn: ListNetworkMigrationCodeGenerations
	mgn: ListNetworkMigrationDefinitions
	mgn: ListNetworkMigrationDeployedStacks
	mgn: ListNetworkMigrationDeployments
	mgn: ListNetworkMigrationExecutions
	mgn: ListNetworkMigrationMapperSegmentConstructs
	mgn: ListNetworkMigrationMapperSegments
	mgn: ListNetworkMigrationMappingUpdates
	mgn: ListNetworkMigrationMappings
	mgn: ListSourceServerActions
	mgn: ListTemplateActions
	mgn: MarkAsArchived
	mgn: PauseReplication
	mgn: PutSourceServerAction
	mgn: PutTemplateAction
	mgn: RemoveSourceServerAction
	mgn: RemoveTemplateAction
	mgn: ResumeReplication
	mgn: RetryDataReplication
	mgn: StartCutover

Awalan layanan	Tindakan
	mgn: StartExport
	mgn: StartImport
	mgn: StartImportFileEnrichment
	mgn: StartNetworkMigrationCodeGeneration
	mgn: StartNetworkMigrationMapping
	mgn: StartNetworkMigrationMappingUpdate
	mgn: StartReplication
	mgn: StartTest
	mgn: StopReplication
	mgn: TerminateTargetInstances
	mgn: UnarchiveApplication
	mgn: UnarchiveWave
	mgn: UpdateApplication
	mgn: UpdateConnector
	mgn: UpdateLaunchConfigurationTemplate
	mgn: UpdateNetworkMigrationDefinition
	mgn: UpdateNetworkMigrationMapperSegment
	mgn: UpdateReplicationConfiguration
	mgn: UpdateReplicationConfigurationTemplate
	mgn: UpdateSourceServer
	mgn: UpdateSourceServerReplicationType

Awalan layanan	Tindakan
	mgn: UpdateWave

Awalan layanan	Tindakan
Migrationhub-strategi	strategi hub migrasi: GetAntiPattern strategi hub migrasi: GetApplicationComponentDetails strategi hub migrasi: GetApplicationComponentStrategies strategi hub migrasi: GetAssessment strategi hub migrasi: GetImportFileTask strategi hub migrasi: GetLatestAssessmentId strategi hub migrasi: GetMessage strategi hub migrasi: GetPortfolioPreferences strategi hub migrasi: GetPortfolioSummary strategi hub migrasi: GetRecommendationReportDetails strategi hub migrasi: GetServerDetails strategi hub migrasi: GetServerStrategies strategi hub migrasi: ListAnalyzableServers strategi hub migrasi: ListAntiPatterns strategi hub migrasi: ListApplicationComponents strategi hub migrasi: ListCollectors strategi hub migrasi: ListImportFileTask strategi hub migrasi: ListJarArtifacts strategi hub migrasi: ListServers strategi hub migrasi: PutLogData strategi hub migrasi: PutMetricData

Awalan layanan	Tindakan
	strategi hub migrasi: PutPortfolioPreferences strategi hub migrasi: RegisterCollector strategi hub migrasi: SendMessage strategi hub migrasi: StartAssessment strategi hub migrasi: StartImportFileTask strategi hub migrasi: StartRecommendationReportGeneration strategi hub migrasi: StopAssessment strategi hub migrasi: UpdateApplicationComponentConfig strategi hub migrasi: UpdateCollectorConfiguration strategi hub migrasi: UpdateServerConfig

Awalan layanan	Tindakan
penargetan seluler	penargetan seluler: CreateApp
	penargetan seluler: CreateCampaign
	penargetan seluler: CreateEmailTemplate
	penargetan seluler: CreateExportJob
	penargetan seluler: CreateImportJob
	penargetan seluler: CreateInAppTemplate
	penargetan seluler: CreateJourney
	penargetan seluler: CreatePushTemplate
	penargetan seluler: CreateRecommenderConfiguration
	penargetan seluler: CreateSegment
	penargetan seluler: CreateSmsTemplate
	penargetan seluler: CreateVoiceTemplate
	penargetan seluler: DeleteAdmChannel
	penargetan seluler: DeleteApnsChannel
	penargetan seluler: DeleteApnsSandboxChannel
	penargetan seluler: DeleteApnsVoipChannel
	penargetan seluler: DeleteApnsVoipSandboxChannel
	penargetan seluler: DeleteApp
	penargetan seluler: DeleteBaiduChannel
	penargetan seluler: DeleteCampaign
	penargetan seluler: DeleteEmailChannel

Awalan layanan	Tindakan
	penargetan seluler: DeleteEmailTemplate
	penargetan seluler: DeleteEndpoint
	penargetan seluler: DeleteEventStream
	penargetan seluler: DeleteGcmChannel
	penargetan seluler: DeleteInAppTemplate
	penargetan seluler: DeleteJourney
	penargetan seluler: DeletePushTemplate
	penargetan seluler: DeleteRecommenderConfiguration
	penargetan seluler: DeleteSegment
	penargetan seluler: DeleteSmsChannel
	penargetan seluler: DeleteSmsTemplate
	penargetan seluler: DeleteUserEndpoints
	penargetan seluler: DeleteVoiceChannel
	penargetan seluler: DeleteVoiceTemplate
	penargetan seluler: GetAdmChannel
	penargetan seluler: GetApnsChannel
	penargetan seluler: GetApnsSandboxChannel
	penargetan seluler: GetApnsVoipChannel
	penargetan seluler: GetApnsVoipSandboxChannel
	penargetan seluler: GetApp
	penargetan seluler: GetApplicationDateRangeKpi

Awalan layanan	Tindakan
	penargetan seluler: GetApplicationSettings
	penargetan seluler: GetApps
	penargetan seluler: GetBaiduChannel
	penargetan seluler: GetCampaign
	penargetan seluler: GetCampaignActivities
	penargetan seluler: GetCampaignDateRangeKpi
	penargetan seluler: GetCampaignVersion
	penargetan seluler: GetCampaignVersions
	penargetan seluler: GetCampaigns
	penargetan seluler: GetChannels
	penargetan seluler: GetEmailChannel
	penargetan seluler: GetEmailTemplate
	penargetan seluler: GetEndpoint
	penargetan seluler: GetEventStream
	penargetan seluler: GetExportJob
	penargetan seluler: GetExportJobs
	penargetan seluler: GetGcmChannel
	penargetan seluler: GetImportJob
	penargetan seluler: GetImportJobs
	penargetan seluler: GetInAppMessages
	penargetan seluler: GetInAppTemplate

Awalan layanan	Tindakan
	penargetan seluler: GetJourney
	penargetan seluler: GetJourneyDateRangeKpi
	penargetan seluler: GetJourneyExecutionActivityMetrics
	penargetan seluler: GetJourneyExecutionMetrics
	penargetan seluler: GetJourneyRunExecutionActivityMetrics
	penargetan seluler: GetJourneyRunExecutionMetrics
	penargetan seluler: GetJourneyRuns
	penargetan seluler: GetPushTemplate
	penargetan seluler: GetRecommenderConfiguration
	penargetan seluler: GetRecommenderConfigurations
	penargetan seluler: GetSegment
	penargetan seluler: GetSegmentExportJobs
	penargetan seluler: GetSegmentImportJobs
	penargetan seluler: GetSegmentVersion
	penargetan seluler: GetSegmentVersions
	penargetan seluler: GetSegments
	penargetan seluler: GetSmsChannel
	penargetan seluler: GetSmsTemplate
	penargetan seluler: GetUserEndpoints
	penargetan seluler: GetVoiceChannel
	penargetan seluler: GetVoiceTemplate

Awalan layanan	Tindakan
	penargetan seluler: ListJourneys
	penargetan seluler: ListTemplateVersions
	penargetan seluler: ListTemplates
	penargetan seluler: PhoneNumberValidate
	penargetan seluler: PutEventStream
	penargetan seluler: RemoveAttributes
	penargetan seluler: UpdateAdmChannel
	penargetan seluler: UpdateApnsChannel
	penargetan seluler: UpdateApnsSandboxChannel
	penargetan seluler: UpdateApnsVoipChannel
	penargetan seluler: UpdateApnsVoipSandboxChannel
	penargetan seluler: UpdateApplicationSettings
	penargetan seluler: UpdateBaiduChannel
	penargetan seluler: UpdateCampaign
	penargetan seluler: UpdateEmailChannel
	penargetan seluler: UpdateEmailTemplate
	penargetan seluler: UpdateEndpoint
	penargetan seluler: UpdateEndpointsBatch
	penargetan seluler: UpdateGcmChannel
	penargetan seluler: UpdateInAppTemplate
	penargetan seluler: UpdateJourney

Awalan layanan	Tindakan
	penargetan seluler: UpdateJourneyState
	penargetan seluler: UpdatePushTemplate
	penargetan seluler: UpdateRecommenderConfiguration
	penargetan seluler: UpdateSegment
	penargetan seluler: UpdateSmsChannel
	penargetan seluler: UpdateSmsTemplate
	penargetan seluler: UpdateTemplateActiveVersion
	penargetan seluler: UpdateVoiceChannel
	penargetan seluler: UpdateVoiceTemplate
	Penargetan Mobile:VerifyotpMessage

Awalan layanan	Tindakan
mq	mq: CreateBroker mq: CreateConfiguration mq: CreateUser mq: DeleteBroker mq: DeleteConfiguration mq: DeleteUser mq: DescribeBroker mq: DescribeBrokerEngineTypes mq: DescribeBrokerInstanceOptions mq: DescribeConfiguration mq: DescribeConfigurationRevision mq: DescribeSharedResources mq: DescribeUser mq: ListBrokers mq: ListConfigurationRevisions mq: ListConfigurations mq: ListUsers MQ: Mempromosikan mq: RebootBroker mq: UpdateBroker mq: UpdateConfiguration

Awalan layanan	Tindakan
	mq: UpdateUser

Awalan layanan	Tindakan
manajer jaringan	manajer jaringan: AcceptAttachment
	manajer jaringan: AssociateConnectPeer
	manajer jaringan: AssociateCustomerGateway
	manajer jaringan: AssociateLink
	manajer jaringan: AssociateTransitGatewayConnectPeer
	manajer jaringan: CreateConnectAttachment
	manajer jaringan: CreateConnectPeer
	manajer jaringan: CreateConnection
	manajer jaringan: CreateCoreNetwork
	manajer jaringan: CreateDevice
	manajer jaringan: CreateDirectConnectGatewayAttachment
	manajer jaringan: CreateGlobalNetwork
	manajer jaringan: CreateLink
	manajer jaringan: CreateSite
	manajer jaringan: CreateSiteToSiteVpnAttachment
	manajer jaringan: CreateTransitGatewayPeering
	manajer jaringan: CreateTransitGatewayRouteTableAttachment
	manajer jaringan: CreateVpcAttachment
	manajer jaringan: DeleteAttachment
	manajer jaringan: DeleteConnectPeer
	manajer jaringan: DeleteConnection

Awalan layanan	Tindakan
	manajer jaringan: DeleteCoreNetwork
	manajer jaringan: DeleteCoreNetworkPolicyVersion
	manajer jaringan: DeleteDevice
	manajer jaringan: DeleteGlobalNetwork
	manajer jaringan: DeleteLink
	manajer jaringan: DeletePeering
	manajer jaringan: DeleteResourcePolicy
	manajer jaringan: DeleteSite
	manajer jaringan: DeregisterTransitGateway
	manajer jaringan: DescribeGlobalNetworks
	manajer jaringan: DisassociateConnectPeer
	manajer jaringan: DisassociateCustomerGateway
	manajer jaringan: DisassociateLink
	manajer jaringan: DisassociateTransitGatewayConnectPeer
	manajer jaringan: ExecuteCoreNetworkChangeSet
	manajer jaringan: GetConnectAttachment
	manajer jaringan: GetConnectPeer
	manajer jaringan: GetConnectPeerAssociations
	manajer jaringan: GetConnections
	manajer jaringan: GetCoreNetwork
	manajer jaringan: GetCoreNetworkChangeEvents

Awalan layanan	Tindakan
	manajer jaringan: GetCoreNetworkChangeSet
	manajer jaringan: GetCoreNetworkPolicy
	manajer jaringan: GetCustomerGatewayAssociations
	manajer jaringan: GetDevices
	manajer jaringan: GetLinkAssociations
	manajer jaringan: GetLinks
	manajer jaringan: GetNetworkResourceCounts
	manajer jaringan: GetNetworkResourceRelationships
	manajer jaringan: GetNetworkResources
	manajer jaringan: GetNetworkRoutes
	manajer jaringan: GetNetworkTelemetry
	manajer jaringan: GetResourcePolicy
	manajer jaringan: GetRouteAnalysis
	manajer jaringan: GetSiteToSiteVpnAttachment
	manajer jaringan: GetSites
	manajer jaringan: GetTransitGatewayConnectPeerAssociations
	manajer jaringan: GetTransitGatewayPeering
	manajer jaringan: GetTransitGatewayRegistrations
	manajer jaringan: GetTransitGatewayRouteTableAttachment
	manajer jaringan: GetVpcAttachment
	manajer jaringan: ListAttachmentRoutingPolicyAssociations

Awalan layanan	Tindakan
	manajer jaringan: ListAttachments
	manajer jaringan: ListConnectPeers
	manajer jaringan: ListCoreNetworkPolicyVersions
	manajer jaringan: ListCoreNetworkPrefixListAssociations
	manajer jaringan: ListCoreNetworkRoutingInformation
	manajer jaringan: ListCoreNetworks
	manajer jaringan: ListOrganizationServiceAccessStatus
	manajer jaringan: ListPeerings
	manajer jaringan: PutAttachmentRoutingPolicyLabel
	manajer jaringan: PutCoreNetworkPolicy
	manajer jaringan: PutResourcePolicy
	manajer jaringan: RegisterTransitGateway
	manajer jaringan: RejectAttachment
	manajer jaringan: RemoveAttachmentRoutingPolicyLabel
	manajer jaringan: RestoreCoreNetworkPolicyVersion
	manajer jaringan: StartOrganizationServiceAccessUpdate
	manajer jaringan: StartRouteAnalysis
	manajer jaringan: UpdateConnection
	manajer jaringan: UpdateCoreNetwork
	manajer jaringan: UpdateDevice
	manajer jaringan: UpdateDirectConnectGatewayAttachment

Awalan layanan	Tindakan
	manajer jaringan: UpdateGlobalNetwork manajer jaringan: UpdateLink manajer jaringan: UpdateNetworkResourceMetadata manajer jaringan: UpdateSite manajer jaringan: UpdateVpcAttachment

Awalan layanan	Tindakan
gesit	gesit: AcceptEulas gesit: CreateLaunchProfile gesit: CreateStreamingImage gesit: CreateStreamingSession gesit: CreateStreamingSessionStream gesit: CreateStudio gesit: CreateStudioComponent gesit: DeleteLaunchProfile gesit: DeleteLaunchProfileMember gesit: DeleteStreamingImage gesit: DeleteStreamingSession gesit: DeleteStudio gesit: DeleteStudioComponent gesit: DeleteStudioMember gesit: GetEula gesit: GetLaunchProfileDetails gesit: GetStreamingImage gesit: GetStreamingSession gesit: GetStreamingSessionBackup gesit: GetStreamingSessionStream gesit: GetStudio

Awalan layanan	Tindakan
	gesit: GetStudioComponent gesit: GetStudioMember gesit: ListEulas gesit: ListLaunchProfileMembers gesit: ListLaunchProfiles gesit: ListStreamingImages gesit: ListStreamingSessionBackups gesit: ListStreamingSessions gesit: ListStudioComponents gesit: ListStudioMembers gesit: ListStudios gesit: PutLaunchProfileMembers gesit: PutStudioMembers gesit: StartStreamingSession gesit: StartStudioSSOConfigurationRepair gesit: StopStreamingSession gesit: UpdateLaunchProfile gesit: UpdateLaunchProfileMember gesit: UpdateStreamingImage gesit: UpdateStudio gesit: UpdateStudioComponent

Awalan layanan	Tindakan
omics	omics: AbortMultipartReadSetUpload
	omics: AcceptShare
	omics: BatchDeleteReadSet
	omics: CancelAnnotationImportJob
	omics: CancelRun
	omics: CancelRunBatch
	omics: CancelVariantImportJob
	omics: CompleteMultipartReadSetUpload
	omics: CreateAnnotationStore
	omics: CreateAnnotationStoreVersion
	omics: CreateConfiguration
	omics: CreateMultipartReadSetUpload
	omics: CreateReferenceStore
	omics: CreateRunGroup
	omics: CreateSequenceStore
	omics: CreateShare
	omics: CreateVariantStore
	omics: CreateWorkflow
	omics: CreateWorkflowVersion
	omics: DeleteAnnotationStore
	omics: DeleteAnnotationStoreVersions

Awalan layanan	Tindakan
	omics: DeleteBatch
	omics: DeleteConfiguration
	omics: DeleteReference
	omics: DeleteReferenceStore
	omics: DeleteRun
	omics: DeleteRunBatch
	omics: DeleteRunGroup
	omics: DeleteSequenceStore
	omics: DeleteShare
	omics: DeleteVariantStore
	omics: DeleteWorkflow
	omics: DeleteWorkflowVersion
	omics: GetAnnotationImportJob
	omics: GetAnnotationStore
	omics: GetAnnotationStoreVersion
	omics: GetBatch
	omics: GetConfiguration
	omics: GetReadSet
	omics: GetReadSetActivationJob
	omics: GetReadSetExportJob
	omics: GetReadSetImportJob

Awalan layanan	Tindakan
	omics: GetReadSetMetadata
	omics: GetReference
	omics: GetReferenceImportJob
	omics: GetReferenceMetadata
	omics: GetReferenceStore
	omics: GetRun
	omics: GetRunGroup
	omics: GetRunTask
	omics: GetSequenceStore
	omics: GetShare
	omics: GetVariantImportJob
	omics: GetVariantStore
	omics: GetWorkflow
	omics: GetWorkflowVersion
	omics: ListAnnotationImportJobs
	omics: ListAnnotationStoreVersions
	omics: ListAnnotationStores
	omics: ListBatch
	omics: ListConfigurations
	omics: ListMultipartReadSetUploads
	omics: ListReadSetActivationJobs

Awalan layanan	Tindakan
	omics: ListReadSetExportJobs
	omics: ListReadSetImportJobs
	omics: ListReadSetUploadParts
	omics: ListReadSets
	omics: ListReferenceImportJobs
	omics: ListReferenceStores
	omics: ListReferences
	omics: ListRunGroups
	omics: ListRunTasks
	omics: ListRuns
	omics: ListRunsInBatch
	omics: ListSequenceStores
	omics: ListShares
	omics: ListVariantImportJobs
	omics: ListVariantStores
	omics: ListWorkflowVersions
	omics: ListWorkflows
	omics: StartAnnotationImportJob
	omics: StartReadSetActivationJob
	omics: StartReadSetExportJob
	omics: StartReadSetImportJob

Awalan layanan	Tindakan
	omics: StartReferenceImportJob
	omics: StartRun
	omics: StartRunBatch
	omics: StartVariantImportJob
	omics: UpdateAnnotationStore
	omics: UpdateAnnotationStoreVersion
	omics: UpdateRunGroup
	omics: UpdateVariantStore
	omics: UpdateWorkflow
	omics: UpdateWorkflowVersion
	omics: UploadReadSetPart

Awalan layanan	Tindakan
opsworks	opsworks: AssignInstance
	opsworks: AssignVolume
	opsworks: AssociateElasticIp
	opsworks: AttachElasticLoadBalancer
	opsworks: CloneStack
	opsworks: CreateApp
	opsworks: CreateDeployment
	opsworks: CreateInstance
	opsworks: CreateLayer
	opsworks: CreateStack
	opsworks: CreateUserProfile
	opsworks: DeleteApp
	opsworks: DeleteInstance
	opsworks: DeleteLayer
	opsworks: DeleteStack
	opsworks: DeleteUserProfile
	opsworks: DeregisterEcsCluster
	opsworks: DeregisterElasticIp
	opsworks: DeregisterInstance
	opsworks: DeregisterRdsDbInstance
	opsworks: DeregisterVolume

Awalan layanan	Tindakan
	opsworks: DescribeAgentVersions opsworks: DescribeApps opsworks: DescribeCommands opsworks: DescribeDeployments opsworks: DescribeEcsClusters opsworks: DescribeElasticIps opsworks: DescribeElasticLoadBalancers opsworks: DescribeInstances opsworks: DescribeLayers opsworks: DescribeLoadBasedAutoScaling opsworks: DescribeMyUserProfile opsworks: DescribeOperatingSystems opsworks: DescribePermissions opsworks: DescribeRaidArrays opsworks: DescribeRdsDbInstances opsworks: DescribeServiceErrors opsworks: DescribeStackProvisioningParameters opsworks: DescribeStackSummary opsworks: DescribeStacks opsworks: DescribeTimeBasedAutoScaling opsworks: DescribeUserProfiles

Awalan layanan	Tindakan
	opsworks: DescribeVolumes opsworks: DetachElasticLoadBalancer opsworks: DisassociateElasticIp opsworks: GetHostnameSuggestion opsworks: GrantAccess opsworks: RebootInstance opsworks: RegisterEcsCluster opsworks: RegisterElasticIp opsworks: RegisterInstance opsworks: RegisterRdsDbInstance opsworks: RegisterVolume opsworks: SetLoadBasedAutoScaling opsworks: SetPermission opsworks: SetTimeBasedAutoScaling opsworks: StartInstance opsworks: StartStack opsworks: StopInstance opsworks: StopStack opsworks: UnassignInstance opsworks: UnassignVolume opsworks: UpdateApp

Awalan layanan	Tindakan
	opsworks: UpdateElasticIp opsworks: UpdateInstance opsworks: UpdateLayer opsworks: UpdateMyUserProfile opsworks: UpdateRdsDbInstance opsworks: UpdateStack opsworks: UpdateUserProfile opsworks: UpdateVolume

Awalan layanan	Tindakan
opsworks-cm	opsworks-cm: AssociateNode
	opsworks-cm: CreateBackup
	opsworks-cm: CreateServer
	opsworks-cm: DeleteBackup
	opsworks-cm: DeleteServer
	opsworks-cm: DescribeAccountAttributes
	opsworks-cm: DescribeBackups
	opsworks-cm: DescribeEvents
	opsworks-cm: DescribeNodeAssociationStatus
	opsworks-cm: DescribeServers
	opsworks-cm: DisassociateNode
	opsworks-cm: ExportServerEngineAttribute
	opsworks-cm: RestoreServer
	opsworks-cm: StartMaintenance
	opsworks-cm: UpdateServer
	opsworks-cm: UpdateServerEngineAttributes

Awalan layanan	Tindakan
organisasi	organisasi: AcceptHandshake organisasi: AttachPolicy organisasi: CancelHandshake organisasi: CloseAccount organisasi: CreateAccount organisasi: CreateGovCloudAccount organisasi: CreateOrganization organisasi: CreateOrganizationalUnit organisasi: CreatePolicy organisasi: DeclineHandshake organisasi: DeleteOrganization organisasi: DeleteOrganizationalUnit organisasi: DeletePolicy organisasi: DeleteResourcePolicy organisasi: DeregisterDelegatedAdministrator organisasi: DescribeAccount organisasi: DescribeCreateAccountStatus organisasi: DescribeEffectivePolicy organisasi: DescribeHandshake organisasi: DescribeOrganization organisasi: DescribeOrganizationalUnit

Awalan layanan	Tindakan
	organisasi: DescribePolicy organisasi: DescribeResourcePolicy organisasi: DescribeResponsibilityTransfer organisasi: DetachPolicy organisasi: DisableAWSServiceAccess organisasi: DisablePolicyType organisasi: EnableAWSServiceAccess organisasi: EnableAllFeatures organisasi: EnablePolicyType organisasi: InviteAccountToOrganization organisasi: LeaveOrganization organisasi: ListAWSServiceAccessForOrganization organisasi: ListAccounts organisasi: ListAccountsForParent organisasi: ListAccountsWithInvalidEffectivePolicy organisasi: ListChildren organisasi: ListCreateAccountStatus organisasi: ListDelegatedAdministrators organisasi: ListDelegatedServicesForAccount organisasi: ListHandshakesForAccount organisasi: ListHandshakesForOrganization

Awalan layanan	Tindakan
	organisasi: ListInboundResponsibilityTransfers organisasi: ListOrganizationalUnitsForParent organisasi: ListOutboundResponsibilityTransfers organisasi: ListParents organisasi: ListPolicies organisasi: ListPoliciesForTarget organisasi: ListRoots organisasi: ListTargetsForPolicy organisasi: MoveAccount organisasi: PutResourcePolicy organisasi: RegisterDelegatedAdministrator organisasi: RemoveAccountFromOrganization organisasi: TerminateResponsibilityTransfer organisasi: UpdateOrganizationalUnit organisasi: UpdatePolicy organisasi: UpdateResponsibilityTransfer

Awalan layanan	Tindakan
pos terdepan	pos terdepan: CancelCapacityTask
	pos terdepan: CancelOrder
	pos terdepan: CreateOrder
	pos terdepan: CreateOutpost
	pos terdepan: CreatePrivateConnectivityConfig
	pos terdepan: CreateQuote
	pos terdepan: CreateRenewal
	pos terdepan: CreateSite
	pos terdepan: DeleteOutpost
	pos terdepan: DeleteQuote
	pos terdepan: DeleteSite
	pos terdepan: GetCapacityTask
	pos terdepan: GetCatalogItem
	pos terdepan: GetConnection
	pos terdepan: GetOrder
	pos terdepan: GetOutpost
	pos terdepan: GetOutpostBillingInformation
	pos terdepan: GetOutpostInstanceTypes
	pos terdepan: GetOutpostSupportedInstanceTypes
	pos terdepan: GetPrivateConnectivityConfig
	pos terdepan: GetQuote

Awalan layanan	Tindakan
	pos terdepan: GetRenewalPricing
	pos terdepan: GetSite
	pos terdepan: GetSiteAddress
	pos terdepan: ListAssetInstances
	pos terdepan: ListAssets
	pos terdepan: ListBlockingInstancesForCapacityTask
	pos terdepan: ListCapacityTasks
	pos terdepan: ListCatalogItems
	pos terdepan: ListOrderableInstanceTypes
	pos terdepan: ListOrders
	pos terdepan: ListOutposts
	pos terdepan: ListQuotes
	pos terdepan: ListSites
	pos terdepan: StartCapacityTask
	pos terdepan: StartConnection
	pos terdepan: UpdateOutpost
	pos terdepan: UpdateQuote
	pos terdepan: UpdateSite
	pos terdepan: UpdateSiteAddress
	pos terdepan: UpdateSiteRackPhysicalProperties

Awalan layanan	Tindakan
pi	pi: CreatePerformanceAnalysisReport pi: DeletePerformanceAnalysisReport pi: DescribeDimensionKeys pi: GetDimensionKeyDetails pi: GetPerformanceAnalysisReport pi: GetResourceMetadata pi: GetResourceMetrics pi: ListAvailableResourceDimensions pi: ListAvailableResourceMetrics pi: ListPerformanceAnalysisReports
pipa	pipa: CreatePipe pipa: DeletePipe pipa: DescribePipe pipa: ListPipes pipa: StartPipe pipa: StopPipe pipa: UpdatePipe

Awalan layanan	Tindakan
polly	polly: DeleteLexicon polly: DescribeVoices polly: GetLexicon polly: GetSpeechSynthesisTask polly: ListLexicons polly: ListSpeechSynthesisTasks polly: PutLexicon polly: StartSpeechSynthesisTask polly: SynthesizeSpeech

Awalan layanan	Tindakan
profile	Profil: AddProfileKey
	Profil: BatchGetCalculatedAttributeForProfile
	Profil: BatchGetProfile
	Profil: CreateCalculatedAttributeDefinition
	Profil: CreateDomain
	Profil: CreateEventStream
	Profil: CreateProfile
	Profil: CreateRecommender
	Profil: CreateRecommenderFilter
	Profil: CreateRecommenderSchema
	Profil: CreateSegmentDefinition
	Profil: CreateSegmentEstimate
	Profil: CreateSegmentSnapshot
	Profil: CreateUploadJob
	Profil: DeleteCalculatedAttributeDefinition
	Profil: DeleteDomain
	Profil: DeleteDomainObjectType
	Profil: DeleteEventStream
	Profil: DeleteIntegration
	Profil: DeleteProfile
	Profil: DeleteProfileKey

Awalan layanan	Tindakan
	Profil: DeleteProfileObject
	Profil: DeleteProfileObjectType
	Profil: DeleteRecommender
	Profil: DeleteRecommenderFilter
	Profil: DeleteRecommenderSchema
	Profil: DeleteSegmentDefinition
	Profil: DeleteWorkflow
	Profil: DetectProfileObjectType
	Profil: GetAutoMergingPreview
	Profil: GetCalculatedAttributeDefinition
	Profil: GetCalculatedAttributeForProfile
	Profil: GetDomain
	Profil: GetDomainObjectType
	Profil: GetEventStream
	Profil: GetIdentityResolutionJob
	Profil: GetIntegration
	Profil: GetMatches
	Profil: GetObjectTypeAttributeStatistics
	Profil: GetProfileObjectType
	Profil: GetProfileObjectTypeTemplate
	Profil: GetProfileRecommendations

Awalan layanan	Tindakan
	Profil: GetRecommender
	Profil: GetRecommenderFilter
	Profil: GetRecommenderSchema
	Profil: GetSegmentDefinition
	Profil: GetSegmentEstimate
	Profil: GetSegmentMembership
	Profil: GetSegmentSnapshot
	Profil: GetSimilarProfiles
	Profil: GetUploadJob
	Profil: GetUploadJobPath
	Profil: GetWorkflow
	Profil: GetWorkflowSteps
	Profil: ListAccountIntegrations
	Profil: ListCalculatedAttributeDefinitions
	Profil: ListCalculatedAttributesForProfile
	Profil: ListDomainLayouts
	Profil: ListDomainObjectTypes
	Profil: ListDomains
	Profil: ListEventStreams
	Profil: ListIdentityResolutionJobs
	Profil: ListIntegrations

Awalan layanan	Tindakan
	Profil: ListObjectTypeAttributeValues
	Profil: ListObjectTypeAttributes
	Profil: ListProfileAttributeValues
	Profil: ListProfileObjectTypeTemplates
	Profil: ListProfileObjectTypes
	Profil: ListProfileObjects
	Profil: ListRecommenderFilters
	Profil: ListRecommenderRecipes
	Profil: ListRecommenderSchemas
	Profil: ListRecommenders
	Profil: ListRuleBasedMatches
	Profil: ListSegmentDefinitions
	Profil: ListUploadJobs
	Profil: ListWorkflows
	Profil: MergeProfiles
	Profil: PutDomainObjectType
	Profil: PutIntegration
	Profil: PutProfileObject
	Profil: PutProfileObjectType
	Profil: SearchProfiles
	Profil: StartRecommender

Awalan layanan	Tindakan
	Profil: StartUploadJob Profil: StopRecommender Profil: StopUploadJob Profil: UpdateCalculatedAttributeDefinition Profil: UpdateDomain Profil: UpdateProfile Profil: UpdateRecommender

Awalan layanan	Tindakan
qldb	qldb: CancelJournalKinesisStream qldb: CreateLedger qldb: DeleteLedger qldb: DescribeJournalKinesisStream qldb: DescribeJournalS3Export qldb: DescribeLedger qldb: ExportJournalToS3 qldb: GetBlock qldb: GetDigest qldb: GetRevision qldb: ListJournalKinesisStreamsForLedger qldb: ListJournalS3Exports qldb: ListJournalS3ExportsForLedger qldb: ListLedgers qldb: StreamJournalToKinesis qldb: UpdateLedger qldb: UpdateLedgerPermissionsMode

Awalan layanan	Tindakan
ram	ram: AcceptResourceShareInvitation
	ram: AssociateResourceShare
	ram: AssociateResourceSharePermission
	ram: CreatePermission
	ram: CreatePermissionVersion
	ram: CreateResourceShare
	ram: DeletePermission
	ram: DeletePermissionVersion
	ram: DeleteResourceShare
	ram: DisassociateResourceShare
	ram: DisassociateResourceSharePermission
	ram: EnableSharingWithAwsOrganization
	ram: GetPermission
	ram: GetResourcePolicies
	ram: GetResourceShareAssociations
	ram: GetResourceShareInvitations
	ram: GetResourceShares
	ram: ListPendingInvitationResources
	ram: ListPermissionAssociations
	ram: ListPermissionVersions
	domba jantan: ListPermissions

Awalan layanan	Tindakan
	domba jantan: ListPrincipals domba jantan: ListReplacePermissionAssociationsWork domba jantan: ListResourceSharePermissions domba jantan: ListResourceTypes domba jantan: ListResources domba jantan: ListSourceAssociations domba jantan: PromotePermissionCreatedFromPolicy domba jantan: PromoteResourceShareCreatedFromPolicy domba jantan: RejectResourceShareInvitation domba jantan: ReplacePermissionAssociations domba jantan: SetDefaultPermissionVersion domba jantan: UpdateResourceShare
rbin	rbin: CreateRule rbin: DeleteRule rbin: GetRule rbin: ListRules rbin: LockRule rbin: UnlockRule rbin: UpdateRule

Awalan layanan	Tindakan
rds	rds: AddRoleToDBCluster
	rds: AddRoleToDBInstance
	rds: AddSourceIdentifierToSubscription
	rds: ApplyPendingMaintenanceAction
	rds: AuthorizeDBSecurityGroupIngress
	RDS: BacktrackDBCluster
	rds: CancelExportTask
	rds: CopyDBClusterParameterGroup
	rds: CopyDBClusterSnapshot
	rds: CopyDBParameterGroup
	RDS: CopyDBSnapshot
	rds: CopyOptionGroup
	rds: CreateCustomDBEngineVersion
	rds: CreateDBClusterParameterGroup
	rds: CreateDBParameterGroup
	RDS: dibuatBProxy
	rds: CreateDBProxyEndpoint
	rds: CreateDBSecurityGroup
	rds: CreateDBSubnetGroup
	rds: CreateEventSubscription
	rds: CreateGlobalCluster

Awalan layanan	Tindakan
	rds: CreateOptionGroup
	rds: DeleteBlueGreenDeployment
	rds: DeleteDBClusterAutomatedBackup
	rds: DeleteDBClusterParameterGroup
	rds: DeleteDBClusterSnapshot
	rds: DeleteDBInstanceAutomatedBackup
	rds: DeleteDBParameterGroup
	RDS: DeletedBProxy
	rds: DeleteDBProxyEndpoint
	rds: DeleteDBSecurityGroup
	RDS: DeletedBSnapshot
	rds: DeleteDBSubnetGroup
	rds: DeleteEventSubscription
	rds: DeleteGlobalCluster
	rds: DeleteOptionGroup
	rds: DeregisterDBProxyTargets
	rds: DescribeAccountAttributes
	rds: DescribeBlueGreenDeployments
	rds: DescribeCertificates
	rds: DescribeDBClusterAutomatedBackups
	rds: DescribeDBClusterBacktracks

Awalan layanan	Tindakan
	rds: DescribeDBClusterEndpoints
	rds: DescribeDBClusterParameterGroups
	rds: DescribeDBClusterParameters
	rds: DescribeDBClusterSnapshotAttributes
	rds: DescribeDBClusterSnapshots
	RDS: DescribedBClusters
	rds: DescribeDBEngineVersions
	rds: DescribeDBInstanceAutomatedBackups
	RDS: DescribedBinstances
	rds: DescribeDBLogFiles
	rds: DescribeDBMajorEngineVersions
	rds: DescribeDBParameterGroups
	RDS: DescribedBParameters
	RDS: DescribedBProxies
	rds: DescribeDBProxyEndpoints
	rds: DescribeDBProxyTargetGroups
	rds: DescribeDBProxyTargets
	RDS:DescribedBrecommendations
	rds: DescribeDBSecurityGroups
	rds: DescribeDBSnapshotAttributes
	rds: DescribeDBSnapshotTenantDatabases

Awalan layanan	Tindakan
	RDS: DescribedBSnapshots
	rds: DescribeDBSubnetGroups
	rds: DescribeEngineDefaultClusterParameters
	rds: DescribeEngineDefaultParameters
	rds: DescribeEventCategories
	rds: DescribeEventSubscriptions
	rds: DescribeEvents
	rds: DescribeExportTasks
	rds: DescribeGlobalClusters
	rds: DescribeIntegrations
	rds: DescribeOptionGroupOptions
	rds: DescribeOptionGroups
	rds: DescribeOrderableDBInstanceOptions
	rds: DescribePendingMaintenanceActions
	rds: DescribeReservedDBInstances
	rds: DescribeReservedDBInstancesOfferings
	rds: DescribeSourceRegions
	rds: DescribeTenantDatabases
	rds: DescribeValidDBInstanceModifications
	rds: DownloadCompleteDBLogFile
	rds: DownloadDBLogFilePortion

Awalan layanan	Tindakan
	RDS: FailoverDBCluster
	rds: FailoverGlobalCluster
	rds: ModifyActivityStream
	rds: ModifyCertificates
	rds: ModifyCurrentDBClusterCapacity
	rds: ModifyDBClusterEndpoint
	rds: ModifyDBClusterParameterGroup
	rds: ModifyDBClusterSnapshotAttribute
	rds: ModifyDBParameterGroup
	RDS: ModifyDBProxy
	rds: ModifyDBProxyEndpoint
	rds: ModifyDBProxyTargetGroup
	RDS: ModifydbreCommendation
	RDS: ModifyDBSnapshot
	rds: ModifyDBSnapshotAttribute
	rds: ModifyDBSubnetGroup
	rds: ModifyEventSubscription
	rds: ModifyGlobalCluster
	rds: ModifyOptionGroup
	rds: ModifyTenantDatabase
	rds: PurchaseReservedDBInstancesOffering

Awalan layanan	Tindakan
	RDS: RebootDBCluster
	rds: RegisterDBProxyTargets
	rds: RemoveFromGlobalCluster
	rds: RemoveRoleFromDBCluster
	rds: RemoveRoleFromDBInstance
	rds: RemoveSourceIdentifierFromSubscription
	rds: ResetDBClusterParameterGroup
	rds: ResetDBParameterGroup
	rds: RestoreDBClusterFromS3
	rds: RestoreDBClusterFromSnapshot
	rds: RestoreDBClusterToPointInTime
	rds: RestoreDBInstanceFromDBSnapshot
	rds: RestoreDBInstanceFromS3
	rds: RestoreDBInstanceToPointInTime
	rds: RevokeDBSecurityGroupIngress
	rds: StartActivityStream
	RDS: StartDBCluster
	RDS: StartDBInstance
	rds: StartDBInstanceAutomatedBackupsReplication
	rds: StartExportTask
	rds: StopActivityStream

Awalan layanan	Tindakan
	RDS: STOPDBCluster RDS: StopdbInstance rds: StopDBInstanceAutomatedBackupsReplication rds: SwitchoverBlueGreenDeployment rds: SwitchoverGlobalCluster rds: SwitchoverReadReplica

Awalan layanan	Tindakan
redshift	pergeseran merah: AcceptReservedNodeExchange
	pergeseran merah: AddPartner
	pergeseran merah: AssociateDataShareConsumer
	pergeseran merah: AuthorizeClusterSecurityGroupIngress
	pergeseran merah: AuthorizeDataShare
	pergeseran merah: AuthorizeEndpointAccess
	pergeseran merah: AuthorizeSnapshotAccess
	pergeseran merah: BatchDeleteClusterSnapshots
	pergeseran merah: BatchModifyClusterSnapshots
	pergeseran merah: CancelResize
	pergeseran merah: CopyClusterSnapshot
	pergeseran merah: CreateAuthenticationProfile
	pergeseran merah: CreateCluster
	pergeseran merah: CreateClusterParameterGroup
	pergeseran merah: CreateClusterSecurityGroup
	pergeseran merah: CreateClusterSnapshot
	pergeseran merah: CreateClusterSubnetGroup
	pergeseran merah: CreateCustomDomainAssociation
	pergeseran merah: CreateEndpointAccess
	pergeseran merah: CreateEventSubscription
	pergeseran merah: CreateHsmClientCertificate

Awalan layanan	Tindakan
	pergeseran merah: CreateHsmConfiguration pergeseran merah: CreateIntegration pergeseran merah: CreateRedshiftIdcApplication pergeseran merah: CreateScheduledAction pergeseran merah: CreateSnapshotCopyGrant pergeseran merah: CreateSnapshotSchedule pergeseran merah: CreateUsageLimit pergeseran merah: DeauthorizeDataShare pergeseran merah: DeleteAuthenticationProfile pergeseran merah: DeleteCluster pergeseran merah: DeleteClusterParameterGroup pergeseran merah: DeleteClusterSecurityGroup pergeseran merah: DeleteClusterSnapshot pergeseran merah: DeleteClusterSubnetGroup pergeseran merah: DeleteCustomDomainAssociation pergeseran merah: DeleteEndpointAccess pergeseran merah: DeleteEventSubscription pergeseran merah: DeleteHsmClientCertificate pergeseran merah: DeleteHsmConfiguration pergeseran merah: DeletePartner pergeseran merah: DeleteRedshiftIdcApplication

Awalan layanan	Tindakan
	pergeseran merah: DeleteResourcePolicy
	pergeseran merah: DeleteScheduledAction
	pergeseran merah: DeleteSnapshotCopyGrant
	pergeseran merah: DeleteSnapshotSchedule
	pergeseran merah: DeleteUsageLimit
	pergeseran merah: DeregisterNamespace
	pergeseran merah: DescribeAccountAttributes
	pergeseran merah: DescribeAuthenticationProfiles
	pergeseran merah: DescribeClusterDbRevisions
	pergeseran merah: DescribeClusterParameterGroups
	pergeseran merah: DescribeClusterParameters
	pergeseran merah: DescribeClusterSecurityGroups
	pergeseran merah: DescribeClusterSnapshots
	pergeseran merah: DescribeClusterSubnetGroups
	pergeseran merah: DescribeClusterTracks
	pergeseran merah: DescribeClusterVersions
	pergeseran merah: DescribeClusters
	pergeseran merah: DescribeCustomDomainAssociations
	pergeseran merah: DescribeDataShares
	pergeseran merah: DescribeDataSharesForConsumer
	pergeseran merah: DescribeDataSharesForProducer

Awalan layanan	Tindakan
	pergeseran merah: DescribeDefaultClusterParameters
	pergeseran merah: DescribeEndpointAccess
	pergeseran merah: DescribeEndpointAuthorization
	pergeseran merah: DescribeEventCategories
	pergeseran merah: DescribeEventSubscriptions
	pergeseran merah: DescribeEvents
	pergeseran merah: DescribeHsmClientCertificates
	pergeseran merah: DescribeHsmConfigurations
	pergeseran merah: DescribeInboundIntegrations
	pergeseran merah: DescribeIntegrations
	pergeseran merah: DescribeLoggingStatus
	pergeseran merah: DescribeNodeConfigurationOptions
	pergeseran merah: DescribeOrderableClusterOptions
	pergeseran merah: DescribePartners
	pergeseran merah: DescribeRedshiftIdcApplications
	pergeseran merah: DescribeReservedNodeExchangeStatus
	pergeseran merah: DescribeReservedNodeOfferings
	pergeseran merah: DescribeReservedNodes
	pergeseran merah: DescribeResize
	pergeseran merah: DescribeScheduledActions
	pergeseran merah: DescribeSnapshotCopyGrants

Awalan layanan	Tindakan
	pergeseran merah: DescribeSnapshotSchedules pergeseran merah: DescribeStorage pergeseran merah: DescribeTableRestoreStatus pergeseran merah: DescribeUsageLimits pergeseran merah: DisableLogging pergeseran merah: DisableSnapshotCopy pergeseran merah: DisassociateDataShareConsumer pergeseran merah: EnableLogging pergeseran merah: EnableSnapshotCopy pergeseran merah: FailoverPrimaryCompute pergeseran merah: GetClusterCredentials pergeseran merah: GetClusterCredentialsWithIAM pergeseran merah: GetIdentityCenterAuthToken pergeseran merah: GetReservedNodeExchangeConfiguration Options pergeseran merah: GetReservedNodeExchangeOfferings pergeseran merah: GetResourcePolicy pergeseran merah: ListRecommendations pergeseran merah: ModifyAquaConfiguration pergeseran merah: ModifyAuthenticationProfile pergeseran merah: ModifyCluster

Awalan layanan	Tindakan
	pergeseran merah: ModifyClusterDbRevision pergeseran merah: ModifyClusterIamRoles pergeseran merah: ModifyClusterMaintenance pergeseran merah: ModifyClusterParameterGroup pergeseran merah: ModifyClusterSnapshot pergeseran merah: ModifyClusterSnapshotSchedule pergeseran merah: ModifyClusterSubnetGroup pergeseran merah: ModifyCustomDomainAssociation pergeseran merah: ModifyEndpointAccess pergeseran merah: ModifyEventSubscription pergeseran merah: ModifyRedshiftIdcApplication pergeseran merah: ModifyScheduledAction pergeseran merah: ModifySnapshotCopyRetentionPeriod pergeseran merah: ModifySnapshotSchedule pergeseran merah: ModifyUsageLimit pergeseran merah: PauseCluster pergeseran merah: PurchaseReservedNodeOffering pergeseran merah: PutResourcePolicy pergeseran merah: RebootCluster pergeseran merah: RegisterNamespace pergeseran merah: RejectDataShare

Awalan layanan	Tindakan
	pergeseran merah: ResetClusterParameterGroup pergeseran merah: ResizeCluster pergeseran merah: RestoreFromClusterSnapshot pergeseran merah: RestoreTableFromClusterSnapshot pergeseran merah: ResumeCluster pergeseran merah: RevokeClusterSecurityGroupIngress pergeseran merah: RevokeEndpointAccess pergeseran merah: RevokeSnapshotAccess pergeseran merah: RotateEncryptionKey pergeseran merah: UpdatePartnerStatus
data pergeseran merah	data pergeseran merah: BatchExecuteStatement data pergeseran merah: CancelStatement data pergeseran merah: DescribeStatement data pergeseran merah: DescribeTable data pergeseran merah: ExecuteStatement data pergeseran merah: GetStatementResult data pergeseran merah: ListDatabases data pergeseran merah: ListSchemas data pergeseran merah: ListSessions data pergeseran merah: ListStatements data pergeseran merah: ListTables

Awalan layanan	Tindakan
ruang refaktor	ruang refaktor: CreateApplication ruang refaktor: CreateEnvironment ruang refaktor: CreateRoute ruang refaktor: CreateService ruang refaktor: DeleteApplication ruang refaktor: DeleteEnvironment ruang refaktor: DeleteResourcePolicy ruang refaktor: DeleteRoute ruang refaktor: DeleteService ruang refaktor: GetApplication ruang refaktor: GetEnvironment ruang refaktor: GetResourcePolicy ruang refaktor: GetRoute ruang refaktor: GetService ruang refaktor: ListApplications ruang refaktor: ListEnvironmentVpcs ruang refaktor: ListEnvironments ruang refaktor: ListRoutes ruang refaktor: ListServices ruang refaktor: PutResourcePolicy ruang refaktor: UpdateRoute

Awalan layanan	Tindakan
rekognisi	Rekognisi: AssociateFaces
	Rekognisi: CompareFaces
	Rekognisi: CopyProjectVersion
	Rekognisi: CreateCollection
	Rekognisi: CreateDataset
	Rekognisi: CreateFaceLivenessSession
	Rekognisi: CreateProject
	Rekognisi: CreateProjectVersion
	Rekognisi: CreateStreamProcessor
	Rekognisi: CreateUser
	Rekognisi: DeleteCollection
	Rekognisi: DeleteDataset
	Rekognisi: DeleteFaces
	Rekognisi: DeleteProject
	Rekognisi: DeleteProjectPolicy
	Rekognisi: DeleteProjectVersion
	Rekognisi: DeleteStreamProcessor
	Rekognisi: DeleteUser
	Rekognisi: DescribeCollection
	Rekognisi: DescribeDataset
	Rekognisi: DescribeProjectVersions

Awalan layanan	Tindakan
	Rekognisi: DescribeProjects
	Rekognisi: DescribeStreamProcessor
	Rekognisi: DetectCustomLabels
	Rekognisi: DetectFaces
	Rekognisi: DetectLabels
	Rekognisi: DetectModerationLabels
	Rekognisi: DetectProtectiveEquipment
	Rekognisi: DetectText
	Rekognisi: DisassociateFaces
	Rekognisi: DistributeDatasetEntries
	Rekognisi: GetCelebrityInfo
	Rekognisi: GetCelebrityRecognition
	Rekognisi: GetContentModeration
	Rekognisi: GetFaceDetection
	Rekognisi: GetFaceLivenessSessionResults
	Rekognisi: GetFaceSearch
	Rekognisi: GetLabelDetection
	Rekognisi: GetMediaAnalysisJob
	Rekognisi: GetPersonTracking
	Rekognisi: GetSegmentDetection
	Rekognisi: GetTextDetection

Awalan layanan	Tindakan
	Rekognisi: IndexFaces
	Rekognisi: ListCollections
	Rekognisi: ListDatasetEntries
	Rekognisi: ListDatasetLabels
	Rekognisi: ListFaces
	Rekognisi: ListMediaAnalysisJobs
	Rekognisi: ListProjectPolicies
	Rekognisi: ListStreamProcessors
	Rekognisi: ListUsers
	Rekognisi: PutProjectPolicy
	Rekognisi: RecognizeCelebrities
	Rekognisi: SearchFaces
	Rekognisi: SearchFacesByImage
	Rekognisi: SearchUsers
	Rekognisi: SearchUsersByImage
	Rekognisi: StartCelebrityRecognition
	Rekognisi: StartContentModeration
	Rekognisi: StartFaceDetection
	Rekognisi: StartFaceLivenessSession
	Rekognisi: StartFaceSearch
	Rekognisi: StartLabelDetection

Awalan layanan	Tindakan
	Rekognisi: StartMediaAnalysisJob
	Rekognisi: StartPersonTracking
	Rekognisi: StartProjectVersion
	Rekognisi: StartSegmentDetection
	Rekognisi: StartStreamProcessor
	Rekognisi: StartTextDetection
	Rekognisi: StopProjectVersion
	Rekognisi: StopStreamProcessor
	Rekognisi: UpdateDatasetEntries
	Rekognisi: UpdateStreamProcessor

Awalan layanan	Tindakan
resiliencehub	resiliencehub: AcceptResourceGroupingRecommendations resiliencehub: AddDraftAppVersionResourceMappings resiliencehub: BatchUpdateRecommendationStatus resiliencehub: CreateApp resiliencehub: CreateAppVersionAppComponent resiliencehub: CreateAppVersionResource resiliencehub: CreateRecommendationTemplate resiliencehub: CreateResiliencyPolicy resiliencehub: DeleteApp resiliencehub: DeleteAppAssessment resiliencehub: DeleteAppInputSource resiliencehub: DeleteAppVersionAppComponent resiliencehub: DeleteAppVersionResource resiliencehub: DeleteRecommendationTemplate resiliencehub: DeleteResiliencyPolicy resiliencehub: DescribeApp resiliencehub: DescribeAppAssessment resiliencehub: DescribeAppVersion resiliencehub: DescribeAppVersionAppComponent resiliencehub: DescribeAppVersionResource resiliencehub: DescribeAppVersionResourcesResolutionStatus

Awalan layanan	Tindakan
	resiliencehub: DescribeAppVersionTemplate resiliencehub: DescribeDraftAppVersionResourcesImportStatus resiliencehub: DescribeMetricsExport resiliencehub: DescribeResiliencyPolicy resiliencehub: DescribeResourceGroupingRecommendationTask resiliencehub: ImportResourcesToDraftAppVersion resiliencehub: ListAlarmRecommendations resiliencehub: ListAppAssessmentComplianceDrifts resiliencehub: ListAppAssessmentResourceDrifts resiliencehub: ListAppAssessments resiliencehub: ListAppComponentCompliances resiliencehub: ListAppComponentRecommendations resiliencehub: ListAppInputSources resiliencehub: ListAppVersionAppComponents resiliencehub: ListAppVersionResourceMappings resiliencehub: ListAppVersionResources resiliencehub: ListAppVersions resiliencehub: ListApps resiliencehub: ListMetrics resiliencehub: ListRecommendationTemplates resiliencehub: ListResiliencyPolicies

Awalan layanan	Tindakan
	resiliencehub: ListResourceGroupingRecommendations resiliencehub: ListSopRecommendations resiliencehub: ListSuggestedResiliencyPolicies resiliencehub: ListTestRecommendations resiliencehub: ListUnsupportedAppVersionResources resiliencehub: PublishAppVersion resiliencehub: PutDraftAppVersionTemplate resiliencehub: RejectResourceGroupingRecommendations resiliencehub: RemoveDraftAppVersionResourceMappings resiliencehub: ResolveAppVersionResources resiliencehub: StartAppAssessment resiliencehub: StartResourceGroupingRecommendationTask resiliencehub: UpdateApp resiliencehub: UpdateAppVersion resiliencehub: UpdateAppVersionAppComponent resiliencehub: UpdateAppVersionResource resiliencehub: UpdateResiliencyPolicy

Awalan layanan	Tindakan
sumber daya-penjelajah-2	sumber daya-penjelajah-2: AssociateDefaultView
	sumber daya-penjelajah-2: BatchGetView
	sumber daya-penjelajah-2: CreateIndex
	sumber daya-penjelajah-2: CreateResourceExplorerSetup
	sumber daya-penjelajah-2: CreateView
	sumber daya-penjelajah-2: DeleteIndex
	sumber daya-penjelajah-2: DeleteResourceExplorerSetup
	sumber daya-penjelajah-2: DeleteView
	sumber daya-penjelajah-2: DisassociateDefaultView
	sumber daya-penjelajah-2: GetAccountLevelServiceConfiguration
	sumber daya-penjelajah-2: GetDefaultView
	sumber daya-penjelajah-2: GetIndex
	sumber daya-penjelajah-2: GetManagedView
	sumber daya-penjelajah-2: GetResourceExplorerSetup
	sumber daya-penjelajah-2: GetServiceIndex
	sumber daya-penjelajah-2: GetServiceView
	sumber daya-penjelajah-2: ListIndexes
	sumber daya-penjelajah-2: ListIndexesForMembers
	sumber daya-penjelajah-2: ListManagedViews
	sumber daya-penjelajah-2: ListServiceIndexes
	sumber daya-penjelajah-2: ListServiceViews

Awalan layanan	Tindakan
	sumber daya-penjelajah-2: ListStreamingAccessForServices
	sumber daya-penjelajah-2: ListSupportedResourceTypes
	sumber daya-penjelajah-2: ListViews
	Resource-Explorer-2:Search
	sumber daya-penjelajah-2: UpdateIndexType
	sumber daya-penjelajah-2: UpdateView

Awalan layanan	Tindakan
kelompok sumber daya	kelompok sumber daya: CancelTagSyncTask kelompok sumber daya: GetAccountSettings kelompok sumber daya: GetGroup kelompok sumber daya: GetGroupConfiguration kelompok sumber daya: GetGroupQuery kelompok sumber daya: GetTagSyncTask kelompok sumber daya: GroupResources kelompok sumber daya: ListGroupResources kelompok sumber daya: ListGroupingStatuses kelompok sumber daya: ListGroups kelompok sumber daya: ListTagSyncTasks kelompok sumber daya: PutGroupConfiguration kelompok sumber daya: SearchResources kelompok sumber daya: StartTagSyncTask kelompok sumber daya: UngroupResources kelompok sumber daya: UpdateAccountSettings kelompok sumber daya: UpdateGroup kelompok sumber daya: UpdateGroupQuery

Awalan layanan	Tindakan
robomaker	pembuat robomak: BatchDeleteWorlds
	pembuat robomak: BatchDescribeSimulationJob
	pembuat robomak: CancelDeploymentJob
	pembuat robomak: CancelSimulationJob
	pembuat robomak: CancelSimulationJobBatch
	pembuat robomak: CancelWorldExportJob
	pembuat robomak: CancelWorldGenerationJob
	pembuat robomak: CreateDeploymentJob
	pembuat robomak: CreateFleet
	pembuat robomak: CreateRobot
	pembuat robomak: CreateRobotApplication
	pembuat robomak: CreateRobotApplicationVersion
	pembuat robomak: CreateSimulationApplication
	pembuat robomak: CreateSimulationApplicationVersion
	pembuat robomak: CreateSimulationJob
	pembuat robomak: CreateWorldExportJob
	pembuat robomak: CreateWorldGenerationJob
	pembuat robomak: CreateWorldTemplate
	pembuat robomak: DeleteFleet
	pembuat robomak: DeleteRobot
	pembuat robomak: DeleteRobotApplication

Awalan layanan	Tindakan
	pembuat robomak: DeleteSimulationApplication
	pembuat robomak: DeleteWorldTemplate
	pembuat robomak: DeregisterRobot
	pembuat robomak: DescribeDeploymentJob
	pembuat robomak: DescribeFleet
	pembuat robomak: DescribeRobot
	pembuat robomak: DescribeRobotApplication
	pembuat robomak: DescribeSimulationApplication
	pembuat robomak: DescribeSimulationJob
	pembuat robomak: DescribeSimulationJobBatch
	pembuat robomak: DescribeWorld
	pembuat robomak: DescribeWorldExportJob
	pembuat robomak: DescribeWorldGenerationJob
	pembuat robomak: DescribeWorldTemplate
	pembuat robomak: GetWorldTemplateBody
	pembuat robomak: ListDeploymentJobs
	pembuat robomak: ListFleets
	pembuat robomak: ListRobotApplications
	pembuat robomak: ListRobots
	pembuat robomak: ListSimulationApplications
	pembuat robomak: ListSimulationJobBatches

Awalan layanan	Tindakan
	pembuat robomak: ListSimulationJobs
	pembuat robomak: ListWorldExportJobs
	pembuat robomak: ListWorldGenerationJobs
	pembuat robomak: ListWorldTemplates
	pembuat robomak: ListWorlds
	pembuat robomak: RegisterRobot
	pembuat robomak: RestartSimulationJob
	pembuat robomak: StartSimulationJobBatch
	pembuat robomak: SyncDeploymentJob
	pembuat robomak: UpdateRobotApplication
	pembuat robomak: UpdateSimulationApplication
	pembuat robomak: UpdateWorldTemplate

Awalan layanan	Tindakan
peranandi mana saja	peranandi mana saja: CreateProfile peranandi mana saja: CreateTrustAnchor peranandi mana saja: DeleteAttributeMapping peranandi mana saja: DeleteCrl peranandi mana saja: DeleteProfile peranandi mana saja: DeleteTrustAnchor peranandi mana saja: DisableCrl peranandi mana saja: DisableProfile peranandi mana saja: DisableTrustAnchor peranandi mana saja: EnableCrl peranandi mana saja: EnableProfile peranandi mana saja: EnableTrustAnchor peranandi mana saja: GetCrl peranandi mana saja: GetProfile peranandi mana saja: GetSubject peranandi mana saja: GetTrustAnchor peranandi mana saja: ImportCrl peranandi mana saja: ListCrls peranandi mana saja: ListProfiles peranandi mana saja: ListSubjects peranandi mana saja: ListTrustAnchors

Awalan layanan	Tindakan
	peranandi mana saja: PutAttributeMapping peranandi mana saja: PutNotificationSettings peranandi mana saja: ResetNotificationSettings peranandi mana saja: UpdateCrl peranandi mana saja: UpdateProfile peranandi mana saja: UpdateTrustAnchor

Awalan layanan	Tindakan
route53	route53: ActivateKeySigningKey
	route53: AssociateVPCWithHostedZone
	route53: ChangeCidrCollection
	route53: ChangeResourceRecordSets
	route53: CreateCidrCollection
	route53: CreateHealthCheck
	route53: CreateHostedZone
	route53: CreateKeySigningKey
	route53: CreateQueryLoggingConfig
	route53: CreateReusableDelegationSet
	route53: CreateTrafficPolicy
	route53: CreateTrafficPolicyInstance
	route53: CreateTrafficPolicyVersion
	route53: CreateVPCAssociationAuthorization
	route53: DeactivateKeySigningKey
	route53: DeleteCidrCollection
	route53: DeleteHealthCheck
	route53: DeleteHostedZone
	route53: DeleteKeySigningKey
	route53: DeleteQueryLoggingConfig
	route53: DeleteReusableDelegationSet

Awalan layanan	Tindakan
	route53: DeleteTrafficPolicy
	route53: DeleteTrafficPolicyInstance
	route53: DeleteVPCAssociationAuthorization
	route53: DisableHostedZoneDNSSEC
	route53: DisassociateVPCFromHostedZone
	route53: EnableHostedZoneDNSSEC
	route53: GetAccountLimit
	route53: GetChange
	route53: GetCheckerIpRanges
	Route53:GetDNSSec
	route53: GetGeoLocation
	route53: GetHealthCheck
	route53: GetHealthCheckCount
	route53: GetHealthCheckLastFailureReason
	route53: GetHealthCheckStatus
	route53: GetHostedZone
	route53: GetHostedZoneCount
	route53: GetHostedZoneLimit
	route53: GetQueryLoggingConfig
	route53: GetReusableDelegationSet
	route53: GetReusableDelegationSetLimit

Awalan layanan	Tindakan
	route53: GetTrafficPolicy
	route53: GetTrafficPolicyInstance
	route53: GetTrafficPolicyInstanceCount
	route53: ListCidrBlocks
	route53: ListCidrCollections
	route53: ListCidrLocations
	route53: ListGeoLocations
	route53: ListHealthChecks
	route53: ListHostedZones
	route53: ListHostedZonesByName
	route53: ListHostedZonesByVPC
	route53: ListQueryLoggingConfigs
	route53: ListResourceRecordSets
	route53: ListReusableDelegationSets
	route53: ListTrafficPolicies
	route53: ListTrafficPolicyInstances
	route53: ListTrafficPolicyInstancesByHostedZone
	route53: ListTrafficPolicyInstancesByPolicy
	route53: ListTrafficPolicyVersions
	route53: ListVPCAssociationAuthorizations
	Route53:testDNSsAnswer

Awalan layanan	Tindakan
	route53: UpdateHealthCheck route53: UpdateHostedZoneComment route53: UpdateTrafficPolicyComment route53: UpdateTrafficPolicyInstance

Awalan layanan	Tindakan
route53-recovery-control-config	route53-recovery-control-config: CreateCluster route53-recovery-control-config: CreateControlPanel route53-recovery-control-config: CreateRoutingControl route53-recovery-control-config: CreateSafetyRule route53-recovery-control-config: DeleteCluster route53-recovery-control-config: DeleteControlPanel route53-recovery-control-config: DeleteRoutingControl route53-recovery-control-config: DeleteSafetyRule route53-recovery-control-config: DescribeCluster route53-recovery-control-config: DescribeControlPanel route53-recovery-control-config: DescribeRoutingControl route53-recovery-control-config: DescribeSafetyRule route53-recovery-control-config: GetResourcePolicy route53-recovery-control-config: ListAssociatedRoute53HealthChecks route53-recovery-control-config: ListClusters route53-recovery-control-config: ListControlPanels route53-recovery-control-config: ListRoutingControls route53-recovery-control-config: ListSafetyRules route53-recovery-control-config: UpdateCluster route53-recovery-control-config: UpdateControlPanel

Awalan layanan	Tindakan
	route53-recovery-control-config: UpdateRoutingControl route53-recovery-control-config: UpdateSafetyRule

Awalan layanan	Tindakan
route53-pemulihan-kesiapan	route53-pemulihan-kesiapan: CreateCell route53-pemulihan-kesiapan: CreateCrossAccountAuthorization route53-pemulihan-kesiapan: CreateReadinessCheck route53-pemulihan-kesiapan: CreateRecoveryGroup route53-pemulihan-kesiapan: CreateResourceSet route53-pemulihan-kesiapan: DeleteCell route53-pemulihan-kesiapan: DeleteCrossAccountAuthorization route53-pemulihan-kesiapan: DeleteReadinessCheck route53-pemulihan-kesiapan: DeleteRecoveryGroup route53-pemulihan-kesiapan: DeleteResourceSet route53-pemulihan-kesiapan: GetArchitectureRecommendations route53-pemulihan-kesiapan: GetCell route53-pemulihan-kesiapan: GetCellReadinessSummary route53-pemulihan-kesiapan: GetReadinessCheck route53-pemulihan-kesiapan: GetReadinessCheckResourceStatus route53-pemulihan-kesiapan: GetReadinessCheckStatus route53-pemulihan-kesiapan: GetRecoveryGroup route53-pemulihan-kesiapan: GetRecoveryGroupReadinessSummary route53-pemulihan-kesiapan: GetResourceSet route53-pemulihan-kesiapan: ListCells

Awalan layanan	Tindakan
	route53-pemulihan-kesiapan: ListCrossAccountAuthorizations
	route53-pemulihan-kesiapan: ListReadinessChecks
	route53-pemulihan-kesiapan: ListRecoveryGroups
	route53-pemulihan-kesiapan: ListResourceSets
	route53-pemulihan-kesiapan: ListRules
	route53-pemulihan-kesiapan: UpdateCell
	route53-pemulihan-kesiapan: UpdateReadinessCheck
	route53-pemulihan-kesiapan: UpdateRecoveryGroup
	route53-pemulihan-kesiapan: UpdateResourceSet

Awalan layanan	Tindakan
route53resolver	route53resolver: AssociateFirewallRuleGroup route53resolver: AssociateResolverEndpointIpAddress route53resolver: AssociateResolverQueryLogConfig route53resolver: AssociateResolverRule route53resolver: CreateFirewallDomainList route53resolver: CreateFirewallRule route53resolver: CreateFirewallRuleGroup route53resolver: CreateResolverEndpoint route53resolver: CreateResolverQueryLogConfig route53resolver: CreateResolverRule route53resolver: DeleteFirewallDomainList route53resolver: DeleteFirewallRule route53resolver: DeleteFirewallRuleGroup route53resolver: DeleteOutpostResolver route53resolver: DeleteResolverEndpoint route53resolver: DeleteResolverQueryLogConfig route53resolver: DeleteResolverRule route53resolver: DisassociateFirewallRuleGroup route53resolver: DisassociateResolverEndpointIpAddress route53resolver: DisassociateResolverQueryLogConfig route53resolver: DisassociateResolverRule

Awalan layanan	Tindakan
	route53resolver: GetFirewallConfig route53resolver: GetFirewallDomainList route53resolver: GetFirewallRuleGroup route53resolver: GetFirewallRuleGroupAssociation route53resolver: GetFirewallRuleGroupPolicy route53resolver: GetOutpostResolver route53resolver: GetResolverConfig route53resolver: GetResolverDnssecConfig route53resolver: GetResolverEndpoint route53resolver: GetResolverQueryLogConfig route53resolver: GetResolverQueryLogConfigAssociation route53resolver: GetResolverQueryLogConfigPolicy route53resolver: GetResolverRule route53resolver: GetResolverRuleAssociation route53resolver: GetResolverRulePolicy route53resolver: ImportFirewallDomains route53resolver: ListFirewallConfigs route53resolver: ListFirewallDomainLists route53resolver: ListFirewallDomains route53resolver: ListFirewallRuleGroupAssociations route53resolver: ListFirewallRuleGroups

Awalan layanan	Tindakan
	route53resolver: ListFirewallRuleTypes route53resolver: ListFirewallRules route53resolver: ListOutpostResolvers route53resolver: ListResolverConfigs route53resolver: ListResolverDnssecConfigs route53resolver: ListResolverEndpointIpAddresses route53resolver: ListResolverEndpoints route53resolver: ListResolverQueryLogConfigAssociations route53resolver: ListResolverQueryLogConfigs route53resolver: ListResolverRuleAssociations route53resolver: ListResolverRules route53resolver: PutFirewallRuleGroupPolicy route53resolver: PutResolverQueryLogConfigPolicy route53resolver: UpdateFirewallConfig route53resolver: UpdateFirewallDomains route53resolver: UpdateFirewallRule route53resolver: UpdateFirewallRuleGroupAssociation route53resolver: UpdateOutpostResolver route53resolver: UpdateResolverConfig route53resolver: UpdateResolverDnssecConfig route53resolver: UpdateResolverEndpoint

Awalan layanan	Tindakan
	route53resolver: UpdateResolverRule
rum	rum: BatchCreateRumMetricDefinitions
	rum: BatchDeleteRumMetricDefinitions
	rum: BatchGetRumMetricDefinitions
	rum: CreateAppMonitor
	rum: DeleteAppMonitor
	rum: DeleteResourcePolicy
	rum: DeleteRumMetricsDestination
	rum: GetAppMonitor
	rum: GetAppMonitorData
	rum: GetResourcePolicy
	rum: ListAppMonitors
	rum: ListRumMetricsDestinations
	rum: PutResourcePolicy
	rum: PutRumMetricsDestination
	rum: UpdateAppMonitor
	rum: UpdateRumMetricDefinition

Awalan layanan	Tindakan
s3	s3: AssociateAccessGrantsIdentityCenter s3: CreateAccessGrant s3: CreateAccessGrantsInstance s3: CreateAccessGrantsLocation s3: CreateAccessPoint s3: CreateAccessPointForObjectLambda s3: CreateBucket s3: CreateBucketMetadataTableConfiguration s3: CreateJob s3: CreateMultiRegionAccessPoint s3: DeleteAccessGrant s3: DeleteAccessGrantsInstance s3: DeleteAccessGrantsInstanceResourcePolicy s3: DeleteAccessGrantsLocation s3: DeleteAccessPoint s3: DeleteAccessPointForObjectLambda s3: DeleteAccessPointPolicy s3: DeleteAccessPointPolicyForObjectLambda s3: DeleteBucket s3: DeleteBucketMetadataTableConfiguration s3: DeleteBucketPolicy

Awalan layanan	Tindakan
	s3: DeleteBucketWebsite s3: DeleteMultiRegionAccessPoint s3: DeleteStorageLensConfiguration s3: DescribeJob s3: DescribeMultiRegionAccessPointOperation s3: DissociateAccessGrantsIdentityCenter s3: GetAccelerateConfiguration s3: GetAccessGrant s3: GetAccessGrantsInstance s3: GetAccessGrantsInstanceForPrefix s3: GetAccessGrantsInstanceResourcePolicy s3: GetAccessGrantsLocation s3: GetAccessPoint s3: GetAccessPointConfigurationForObjectLambda s3: GetAccessPointForObjectLambda s3: GetAccessPointPolicy s3: GetAccessPointPolicyForObjectLambda s3: GetAccessPointPolicyStatus s3: GetAccessPointPolicyStatusForObjectLambda s3: GetAccountPublicAccessBlock s3: GetAnalyticsConfiguration

Awalan layanan	Tindakan
	s3: GetBucketAbac s3: GetBucketAcl s3: GetBucket CORS s3: GetBucketLocation s3: GetBucketLogging s3: GetBucketNotification s3: GetBucketObjectLockConfiguration s3: GetBucketOwnershipControls s3: GetBucketPolicy s3: GetBucketPolicyStatus s3: GetBucketPublicAccessBlock s3: GetBucketRequestPayment s3: GetBucketVersioning s3: GetBucketWebsite s3: GetDataAccess s3: GetEncryptionConfiguration s3: GetIntelligentTieringConfiguration s3: GetInventoryConfiguration s3: GetLifecycleConfiguration s3: GetMetricsConfiguration s3: GetMultiRegionAccessPoint

Awalan layanan	Tindakan
	s3: GetMultiRegionAccessPointPolicy s3: GetMultiRegionAccessPointPolicyStatus s3: GetMultiRegionAccessPointRoutes s3: GetReplicationConfiguration s3: GetStorageLensConfiguration s3: GetStorageLensDashboard s3: ListAccessGrants s3: ListAccessGrantsInstances s3: ListAccessGrantsLocations s3: ListAccessPoints s3: ListAccessPointsForObjectLambda s3: ListAllMyBuckets s3: ListBucketMultipartUploads s3: ListCallerAccessGrants s3: ListJobs s3: ListMultiRegionAccessPoints s3: ListStorageLensConfigurations s3: PutAccelerateConfiguration s3: PutAccessGrantsInstanceResourcePolicy s3: PutAccessPointConfigurationForObjectLambda s3: PutAccessPointPolicy

Awalan layanan	Tindakan
	s3: PutAccessPointPolicyForObjectLambda s3: PutAccountPublicAccessBlock s3: PutAnalyticsConfiguration s3: PutBucketAbac s3: PutBucketAcl s3: PutBucket CORS s3: PutBucketLogging s3: PutBucketNotification s3: PutBucketObjectLockConfiguration s3: PutBucketOwnershipControls s3: PutBucketPolicy s3: PutBucketPublicAccessBlock s3: PutBucketRequestPayment s3: PutBucketVersioning s3: PutBucketWebsite s3: PutEncryptionConfiguration s3: PutIntelligentTieringConfiguration s3: PutInventoryConfiguration s3: PutLifecycleConfiguration s3: PutMetricsConfiguration s3: PutMultiRegionAccessPointPolicy

Awalan layanan	Tindakan
	s3: PutReplicationConfiguration s3: PutStorageLensConfiguration s3: SubmitMultiRegionAccessPointRoutes s3: UpdateAccessGrantsLocation s3: UpdateBucketMetadataAnnotationTableConfiguration s3: UpdateBucketMetadataJournalTableConfiguration s3: UpdateJobPriority s3: UpdateJobStatus
pos terdepan s3	pos terdepan s3: CreateEndpoint pos terdepan s3: DeleteEndpoint pos terdepan s3: ListEndpoints pos terdepan s3: ListOutpostsWithS3 pos terdepan s3: ListSharedEndpoints

Awalan layanan	Tindakan
sagemaker-geospasial	sagemaker-geospasial: DeleteEarthObservationJob sagemaker-geospasial: DeleteVectorEnrichmentJob sagemaker-geospasial: ExportEarthObservationJob sagemaker-geospasial: ExportVectorEnrichmentJob sagemaker-geospasial: GetEarthObservationJob sagemaker-geospasial: GetRasterDataCollection sagemaker-geospasial: GetTile sagemaker-geospasial: GetVectorEnrichmentJob sagemaker-geospasial: ListEarthObservationJobs sagemaker-geospasial: ListRasterDataCollections sagemaker-geospasial: ListVectorEnrichmentJobs sagemaker-geospasial: SearchRasterDataCollection sagemaker-geospasial: StartEarthObservationJob sagemaker-geospasial: StartVectorEnrichmentJob sagemaker-geospasial: StopEarthObservationJob sagemaker-geospasial: StopVectorEnrichmentJob

Awalan layanan	Tindakan
savingsplans	savingsplans: CreateSavingsPlan
	savingsplans: DeleteQueuedSavingsPlan
	savingsplans: DescribeSavingsPlanRates
	savingsplans: DescribeSavingsPlans
	savingsplans: DescribeSavingsPlansOfferingRates
	savingsplans: DescribeSavingsPlansOfferings
	savingsplans: ReturnSavingsPlan

Awalan layanan	Tindakan
skema	skema: CreateDiscoverer skema: CreateRegistry skema: CreateSchema skema: DeleteDiscoverer skema: DeleteRegistry skema: DeleteResourcePolicy skema: DeleteSchema skema: DeleteSchemaVersion skema: DescribeCodeBinding skema: DescribeDiscoverer skema: DescribeRegistry skema: DescribeSchema skema: ExportSchema skema: GetCodeBindingSource skema: GetDiscoveredSchema skema: GetResourcePolicy skema: ListDiscoverers skema: ListRegistries skema: ListSchemaVersions skema: ListSchemas skema: PutCodeBinding

Awalan layanan	Tindakan
	skema: PutResourcePolicy skema: SearchSchemas skema: StartDiscoverer skema: StopDiscoverer skema: UpdateDiscoverer skema: UpdateRegistry skema: UpdateSchema
sdb	sdb: CreateDomain sdb: DeleteDomain sdb: DomainMetadata sdb: ListDomains

Awalan layanan	Tindakan
manajer rahasia	manajer rahasia: CancelRotateSecret manajer rahasia: CreateSecret manajer rahasia: DeleteResourcePolicy manajer rahasia: DeleteSecret manajer rahasia: DescribeSecret manajer rahasia: GetRandomPassword manajer rahasia: GetResourcePolicy manajer rahasia: GetSecretValue manajer rahasia: ListSecretVersionIds manajer rahasia: ListSecrets manajer rahasia: PutResourcePolicy manajer rahasia: PutSecretValue manajer rahasia: RemoveRegionsFromReplication manajer rahasia: ReplicateSecretToRegions manajer rahasia: RestoreSecret manajer rahasia: RotateSecret manajer rahasia: StopReplicationToReplica manajer rahasia: UpdateSecret manajer rahasia: ValidateResourcePolicy

Awalan layanan	Tindakan
securityhub	securityhub: AcceptAdministratorInvitation
	securityhub: AcceptInvitation
	securityhub: BatchDeleteAutomationRules
	securityhub: BatchDisableStandards
	securityhub: BatchEnableStandards
	securityhub: BatchGetAutomationRules
	securityhub: BatchGetConfigurationPolicyAssociations
	securityhub: BatchGetEnabledRegionsV2
	securityhub: BatchGetSecurityControls
	securityhub: BatchGetStandardsControlAssociations
	securityhub: BatchImportFindings
	securityhub: BatchUpdateAutomationRules
	securityhub: BatchUpdateFindings
	securityhub: BatchUpdateStandardsControlAssociations
	securityhub: V2 ConnectorRegistrations
	securityhub: CreateActionTarget
	securityhub: V2 CreateAggregator
	securityhub: CreateAutomationRule
	securityhub: CreateAutomationRuleV2
	securityhub: CreateConfigurationPolicy
	securityhub: CreateConnector

Awalan layanan	Tindakan
	securityhub: V2 CreateConnector
	securityhub: CreateFindingAggregator
	securityhub: CreateInsight
	securityhub: CreateMembers
	securityhub: V2 CreateTicket
	securityhub: DeclineInvitations
	securityhub: DeleteActionTarget
	securityhub: V2 DeleteAggregator
	securityhub: DeleteAutomationRuleV2
	securityhub: DeleteConfigurationPolicy
	securityhub: DeleteConnector
	securityhub: V2 DeleteConnector
	securityhub: DeleteFindingAggregator
	securityhub: DeleteInsight
	securityhub: DeleteInvitations
	securityhub: DeleteMembers
	securityhub: DescribeActionTargets
	securityhub: DescribeHub
	securityhub: DescribeOrganizationConfiguration
	securityhub: DescribeProducts
	securityhub: DescribeSecurityHubV2

Awalan layanan	Tindakan
	securityhub: DescribeStandards
	securityhub: DisableImportFindingsForProduct
	securityhub: DisableOrganizationAdminAccount
	securityhub: DisableSecurityHub
	securityhub: DisableSecurityHubFeatureV2
	securityhub: DisableSecurityHubV2
	securityhub: DisassociateFromAdministratorAccount
	securityhub: DisassociateFromMasterAccount
	securityhub: DisassociateMembers
	securityhub: EnableImportFindingsForProduct
	securityhub: EnableOrganizationAdminAccount
	securityhub: EnableSecurityHub
	securityhub: EnableSecurityHubFeatureV2
	securityhub: GetAdministratorAccount
	securityhub: V2 GetAggregator
	securityhub: GetAutomationRuleV2
	securityhub: GetConfigurationPolicy
	securityhub: GetConfigurationPolicyAssociation
	securityhub: GetConnector
	securityhub: V2 GetConnector
	securityhub: GetCoverageStatisticsV2

Awalan layanan	Tindakan
	securityhub: GetEnabledStandards
	securityhub: GetFindingAggregator
	securityhub: GetFindingHistory
	securityhub: GetFindings
	securityhub: GetInsightResults
	securityhub: GetInsights
	securityhub: GetInvitationsCount
	securityhub: GetMasterAccount
	securityhub: GetMembers
	securityhub: GetSecurityControlDefinition
	securityhub: InviteMembers
	securityhub: V2 ListAggregators
	securityhub: ListAutomationRules
	securityhub: ListAutomationRulesV2
	securityhub: ListConfigurationPolicies
	securityhub: ListConfigurationPolicyAssociations
	securityhub: ListConnectors
	securityhub: V2 ListConnectors
	securityhub: ListEnabledProductsForImport
	securityhub: ListFindingAggregators
	securityhub: ListInvitations

Awalan layanan	Tindakan
	securityhub: ListMembers
	securityhub: ListOrganizationAdminAccounts
	securityhub: ListSecurityControlDefinitions
	securityhub: ListStandardsControlAssociations
	securityhub: StartConfigurationPolicyAssociation
	securityhub: StartConfigurationPolicyDisassociation
	securityhub: UpdateActionTarget
	securityhub: V2 UpdateAggregator
	securityhub: UpdateAutomationRuleV2
	securityhub: UpdateConfigurationPolicy
	securityhub: UpdateConnector
	securityhub: V2 UpdateConnector
	securityhub: UpdateFindingAggregator
	securityhub: UpdateFindings
	securityhub: UpdateInsight
	securityhub: UpdateOrganizationConfiguration
	securityhub: UpdateSecurityControl
	securityhub: UpdateSecurityHubConfiguration

Awalan layanan	Tindakan
Securitylake	Securitylake: CreateAwsLogSource Securitylake: CreateCustomLogSource Securitylake: CreateDataLakeExceptionSubscription Securitylake: CreateDataLakeOrganizationConfiguration Securitylake: CreateSubscriber Securitylake: CreateSubscriberNotification Securitylake: DeleteAwsLogSource Securitylake: DeleteCustomLogSource Securitylake: DeleteDataLakeExceptionSubscription Securitylake: DeleteDataLakeOrganizationConfiguration Securitylake: DeleteSubscriber Securitylake: DeleteSubscriberNotification Securitylake: DeregisterDataLakeDelegatedAdministrator Securitylake: GetDataLakeExceptionSubscription Securitylake: GetDataLakeOrganizationConfiguration Securitylake: GetDataLakeSources Securitylake: GetSubscriber Securitylake: ListDataLakes Securitylake: ListLogSources Securitylake: ListSubscribers Securitylake: RegisterDataLakeDelegatedAdministrator

Awalan layanan	Tindakan
	Securitylake: UpdateDataLakeExceptionSubscription Securitylake: UpdateSubscriber Securitylake: UpdateSubscriberNotification
repo tanpa server	repo tanpa server: CreateApplication repo tanpa server: CreateApplicationVersion repo tanpa server: CreateCloudFormationChangeSet repo tanpa server: CreateCloudFormationTemplate repo tanpa server: DeleteApplication repo tanpa server: GetApplication repo tanpa server: GetApplicationPolicy repo tanpa server: GetCloudFormationTemplate repo tanpa server: ListApplicationDependencies repo tanpa server: ListApplicationVersions repo tanpa server: ListApplications repo tanpa server: PutApplicationPolicy repo tanpa server: UnshareApplication repo tanpa server: UpdateApplication

Awalan layanan	Tindakan
servicecatalog	servicecatalog: AcceptPortfolioShare servicecatalog: AssociateBudgetWithResource servicecatalog: AssociatePrincipalWithPortfolio servicecatalog: AssociateProductWithPortfolio servicecatalog: AssociateServiceActionWithProvisioningArtifact servicecatalog: BatchAssociateServiceActionWithProvisioningArtifact servicecatalog: BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog: CopyProduct servicecatalog: CreateAttributeGroup servicecatalog: CreateConstraint servicecatalog: CreatePortfolio servicecatalog: CreatePortfolioShare servicecatalog: CreateProduct servicecatalog: CreateProvisionedProductPlan servicecatalog: CreateProvisioningArtifact servicecatalog: CreateServiceAction servicecatalog: DeleteAttributeGroup servicecatalog: DeleteConstraint servicecatalog: DeletePortfolio servicecatalog: DeletePortfolioShare

Awalan layanan	Tindakan
	servicecatalog: DeleteProduct servicecatalog: DeleteProvisionedProductPlan servicecatalog: DeleteProvisioningArtifact servicecatalog: DeleteServiceAction servicecatalog: DescribeConstraint servicecatalog: DescribeCopyProductStatus servicecatalog: DescribePortfolio servicecatalog: DescribePortfolioShareStatus servicecatalog: DescribePortfolioShares servicecatalog: DescribeProduct servicecatalog: DescribeProductAsAdmin servicecatalog: DescribeProductView servicecatalog: DescribeProvisionedProduct servicecatalog: DescribeProvisionedProductPlan servicecatalog: DescribeProvisioningArtifact servicecatalog: DescribeProvisioningParameters servicecatalog: DescribeRecord servicecatalog: DescribeServiceAction servicecatalog: DescribeServiceActionExecutionParameters servicecatalog: DisableAWSOrganizationsAccess servicecatalog: DisassociateBudgetFromResource

Awalan layanan	Tindakan
	servicecatalog: DisassociatePrincipalFromPortfolio
	servicecatalog: DisassociateProductFromPortfolio
	servicecatalog: DisassociateServiceActionFromProvisioningArtifact
	servicecatalog: EnableAWSOrganizationsAccess
	servicecatalog: ExecuteProvisionedProductPlan
	servicecatalog: ExecuteProvisionedProductServiceAction
	servicecatalog: GetAWSOrganizationsAccessStatus
	servicecatalog: GetProvisionedProductOutputs
	servicecatalog: ImportAsProvisionedProduct
	servicecatalog: ListAcceptedPortfolioShares
	servicecatalog: ListAttributeGroups
	servicecatalog: ListBudgetsForResource
	servicecatalog: ListConstraintsForPortfolio
	servicecatalog: ListLaunchPaths
	servicecatalog: ListOrganizationPortfolioAccess
	servicecatalog: ListPortfolioAccess
	servicecatalog: ListPortfolios
	servicecatalog: ListPortfoliosForProduct
	servicecatalog: ListPrincipalsForPortfolio
	servicecatalog: ListProvisionedProductPlans
	servicecatalog: ListProvisioningArtifacts

Awalan layanan	Tindakan
	servicecatalog: ListProvisioningArtifactsForServiceAction servicecatalog: ListRecordHistory servicecatalog: ListServiceActions servicecatalog: ListServiceActionsForProvisioningArtifact servicecatalog: ListStackInstancesForProvisionedProduct servicecatalog: NotifyProvisionProductEngineWorkflowResult servicecatalog: NotifyTerminateProvisionedProductEngineWorkflowResult servicecatalog: NotifyUpdateProvisionedProductEngineWorkflowResult servicecatalog: ProvisionProduct servicecatalog: RejectPortfolioShare servicecatalog: ScanProvisionedProducts servicecatalog: SearchProducts servicecatalog: SearchProductsAsAdmin servicecatalog: SearchProvisionedProducts servicecatalog: TerminateProvisionedProduct servicecatalog: UpdateConstraint servicecatalog: UpdatePortfolio servicecatalog: UpdatePortfolioShare servicecatalog: UpdateProduct servicecatalog: UpdateProvisionedProduct

Awalan layanan	Tindakan
	servicecatalog: UpdateProvisionedProductProperties servicecatalog: UpdateProvisioningArtifact servicecatalog: UpdateServiceAction

Awalan layanan	Tindakan
servicediscovery	penemuan layanan: CreateHttpNamespace
	penemuan layanan: CreatePrivateDnsNamespace
	penemuan layanan: CreatePublicDnsNamespace
	penemuan layanan: CreateService
	penemuan layanan: DeleteNamespace
	penemuan layanan: DeleteService
	penemuan layanan: DeleteServiceAttributes
	penemuan layanan: DeregisterInstance
	penemuan layanan: GetInstance
	penemuan layanan: GetInstancesHealthStatus
	penemuan layanan: GetNamespace
	penemuan layanan: GetOperation
	penemuan layanan: GetService
	penemuan layanan: ListInstances
	penemuan layanan: ListNamespaces
	penemuan layanan: ListOperations
	penemuan layanan: ListServices
	penemuan layanan: RegisterInstance
	penemuan layanan: UpdateHttpNamespace
	penemuan layanan: UpdateInstanceCustomHealthStatus
	penemuan layanan: UpdatePrivateDnsNamespace

Awalan layanan	Tindakan
	penemuan layanan: UpdatePublicDnsNamespace penemuan layanan: UpdateService penemuan layanan: UpdateServiceAttributes

Awalan layanan	Tindakan
servicequotas	servicequotas: AssociateServiceQuotaTemplate servicequotas: CreateSupportCase servicequotas: DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas: DisassociateServiceQuotaTemplate servicequotas: GetAWSDefaultServiceQuota servicequotas: GetAssociationForServiceQuotaTemplate servicequotas: GetAutoManagementConfiguration servicequotas: GetQuotaUtilizationReport servicequotas: GetRequestedServiceQuotaChange servicequotas: GetServiceQuota servicequotas: GetServiceQuotaIncreaseRequestFromTemplate servicequotas: ListAWSDefaultServiceQuotas servicequotas: ListRequestedServiceQuotaChangeHistory servicequotas: ListRequestedServiceQuotaChangeHistoryByQuota servicequotas: ListServiceQuotaIncreaseRequestsInTemplate servicequotas: ListServiceQuotas servicequotas: ListServices servicequotas: PutServiceQuotaIncreaseRequestIntoTemplate servicequotas: RequestServiceQuotaIncrease servicequotas: StartAutoManagement servicequotas: StartQuotaUtilizationReport

Awalan layanan	Tindakan
	servicequotas: StopAutoManagement servicequotas: UpdateAutoManagement

Awalan layanan	Tindakan
ses	ses: BatchGetMetricData
	ses: CloneReceiptRuleSet
	ses: CreateAddonInstance
	ses: CreateAddonSubscription
	ses: CreateAddressList
	ses: CreateAddressListImportJob
	ses: CreateArchive
	ses: CreateConfigurationSet
	ses: CreateConfigurationSetEventDestination
	ses: CreateConfigurationSetTrackingOptions
	ses: CreateContact
	ses: CreateContactList
	ses: CreateCustomVerificationEmailTemplate
	ses: CreateDedicatedIpPool
	ses: CreateDeliverabilityTestReport
	ses: CreateEmailIdentity
	ses: CreateEmailIdentityPolicy
	ses: CreateEmailTemplate
	ses: CreateImportJob
	ses: CreateIngressPoint
	ses: CreateMultiRegionEndpoint

Awalan layanan	Tindakan
	ses: CreateReceiptFilter
	ses: CreateReceiptRule
	ses: CreateReceiptRuleSet
	ses: CreateRelay
	ses: CreateRuleSet
	ses: CreateTemplate
	ses: CreateTenant
	ses: CreateTenantResourceAssociation
	ses: CreateTrafficPolicy
	ses: DeleteAddonInstance
	ses: DeleteAddonSubscription
	ses: DeleteAddressList
	ses: DeleteArchive
	ses: DeleteConfigurationSet
	ses: DeleteConfigurationSetEventDestination
	ses: DeleteConfigurationSetTrackingOptions
	ses: DeleteContact
	ses: DeleteContactList
	ses: DeleteCustomVerificationEmailTemplate
	ses: DeleteDedicatedIpPool
	ses: DeleteEmailIdentity

Awalan layanan	Tindakan
	ses: DeleteEmailIdentityPolicy
	ses: DeleteEmailTemplate
	ses: DeleteIdentity
	ses: DeleteIdentityPolicy
	ses: DeleteIngressPoint
	ses: DeleteMultiRegionEndpoint
	ses: DeleteReceiptFilter
	ses: DeleteReceiptRule
	ses: DeleteReceiptRuleSet
	ses: DeleteRelay
	ses: DeleteRuleSet
	ses: DeleteSuppressedDestination
	ses: DeleteTemplate
	ses: DeleteTenant
	ses: DeleteTenantResourceAssociation
	ses: DeleteTrafficPolicy
	ses: DeleteVerifiedEmailAddress
	ses: DeregisterMemberFromAddressList
	ses: DescribeActiveReceiptRuleSet
	ses: DescribeConfigurationSet
	ses: DescribeReceiptRule

Awalan layanan	Tindakan
	ses: DescribeReceiptRuleSet
	ses: GetAccount
	ses: GetAccountSendingEnabled
	ses: GetAddonInstance
	ses: GetAddonSubscription
	ses: GetAddressList
	ses: GetArchive
	ses: GetArchiveExport
	ses: GetArchiveMessage
	ses: GetArchiveMessageContent
	ses: GetArchiveSearch
	ses: GetArchiveSearchResults
	ses: GetBlacklistReports
	ses: GetConfigurationSet
	ses: GetConfigurationSetEventDestinations
	ses: GetContact
	ses: GetContactList
	ses: GetCustomVerificationEmailTemplate
	ses: GetDedicatedIp
	ses: GetDedicatedIpPool
	ses: GetDedicatedIps

Awalan layanan	Tindakan
	ses: GetDeliverabilityDashboardOptions
	ses: GetDeliverabilityTestReport
	ses: GetDomainDeliverabilityCampaign
	ses: GetDomainStatisticsReport
	ses: GetEmailAddressInsights
	ses: GetEmailIdentity
	ses: GetEmailIdentityPolicies
	ses: GetEmailTemplate
	ses: GetIdentityDkimAttributes
	ses: GetIdentityMailFromDomainAttributes
	ses: GetIdentityNotificationAttributes
	ses: GetIdentityPolicies
	ses: GetIdentityVerificationAttributes
	ses: GetImportJob
	ses: GetIngressPoint
	ses: GetMemberOfAddressList
	ses: GetMessageInsights
	ses: GetMultiRegionEndpoint
	ses: GetRelay
	ses: GetRuleSet
	ses: GetSendQuota

Awalan layanan	Tindakan
	ses: GetSendStatistics
	ses: GetSuppressedDestination
	ses: GetTemplate
	ses: GetTenant
	ses: GetTrafficPolicy
	ses: ListAddonInstances
	ses: ListAddonSubscriptions
	ses: ListAddressListImportJobs
	ses: ListAddressLists
	ses: ListArchiveExports
	ses: ListArchiveSearches
	ses: ListArchives
	ses: ListConfigurationSets
	ses: ListContactLists
	ses: ListContacts
	ses: ListCustomVerificationEmailTemplates
	ses: ListDedicatedIpPools
	ses: ListDeliverabilityTestReports
	ses: ListDomainDeliverabilityCampaigns
	ses: ListEmailIdentities
	ses: ListEmailTemplates

Awalan layanan	Tindakan
	ses: ListExportJobs
	ses: ListIdentities
	ses: ListIdentityPolicies
	ses: ListImportJobs
	ses: ListIngressPoints
	ses: ListMembersOfAddressList
	ses: ListMultiRegionEndpoints
	ses: ListReceiptFilters
	ses: ListReceiptRuleSets
	ses: ListRecommendations
	ses: ListRelays
	ses: ListReputationEntities
	ses: ListResourceTenants
	ses: ListRuleSets
	ses: ListSuppressedDestinations
	ses: ListTemplates
	ses: ListTenantResources
	ses: ListTenants
	ses: ListTrafficPolicies
	ses: ListVerifiedEmailAddresses
	ses: PutAccountDedicatedIpWarmupAttributes

Awalan layanan	Tindakan
	ses: PutAccountDetails
	ses: PutAccountSendingAttributes
	ses: PutAccountSuppressionAttributes
	ses: PutAccountVdmAttributes
	ses: PutConfigurationSetArchivingOptions
	ses: PutConfigurationSetDeliveryOptions
	ses: PutConfigurationSetReputationOptions
	ses: PutConfigurationSetSendingOptions
	ses: PutConfigurationSetSuppressionOptions
	ses: PutConfigurationSetTrackingOptions
	ses: PutConfigurationSetVdmOptions
	ses: PutDedicatedIpInPool
	ses: PutDedicatedIpPoolScalingAttributes
	ses: PutDedicatedIpWarmupAttributes
	ses: PutDeliverabilityDashboardOption
	ses: PutEmailIdentityConfigurationSetAttributes
	ses: PutEmailIdentityDkimAttributes
	ses: PutEmailIdentityDkimSigningAttributes
	ses: PutEmailIdentityFeedbackAttributes
	ses: PutEmailIdentityMailFromAttributes
	ses: PutIdentityPolicy

Awalan layanan	Tindakan
	ses: PutSuppressedDestination
	ses: PutTenantSuppressionAttributes
	ses: RegisterMemberToAddressList
	ses: ReorderReceiptRuleSet
	ses: SendBounce
	ses: SendCustomVerificationEmail
	ses: SetActiveReceiptRuleSet
	ses: SetIdentityDkimEnabled
	ses: SetIdentityFeedbackForwardingEnabled
	ses: SetIdentityHeadersInNotificationsEnabled
	ses: SetIdentityMailFromDomain
	ses: SetIdentityNotificationTopic
	ses: SetReceiptRulePosition
	ses: StartArchiveExport
	ses: StartArchiveSearch
	ses: StopArchiveExport
	ses: StopArchiveSearch
	ses: TestRenderEmailTemplate
	ses: TestRenderTemplate
	ses: UpdateAccountSendingEnabled
	ses: UpdateArchive

Awalan layanan	Tindakan
	ses: UpdateConfigurationSetEventDestination
	ses: UpdateConfigurationSetReputationMetricsEnabled
	ses: UpdateConfigurationSetSendingEnabled
	ses: UpdateConfigurationSetTrackingOptions
	ses: UpdateContact
	ses: UpdateContactList
	ses: UpdateCustomVerificationEmailTemplate
	ses: UpdateEmailIdentityPolicy
	ses: UpdateEmailTemplate
	ses: UpdateIngressPoint
	ses: UpdateReceiptRule
	ses: UpdateRelay
	ses: UpdateRuleSet
	ses: UpdateTemplate
	ses: UpdateTrafficPolicy
	ses: VerifyDomainDkim
	ses: VerifyDomainIdentity
	ses: VerifyEmailAddress
	ses: VerifyEmailIdentity

Awalan layanan	Tindakan
melindungi	perisai: AssociateDRTLogBucket perisai: AssociateHealthCheck perisai: AssociateProactiveEngagementDetails perisai: CreateProtection perisai: CreateProtectionGroup perisai: CreateSubscription perisai: DeleteProtection perisai: DeleteProtectionGroup perisai: DeleteSubscription perisai: DescribeAttack perisai: DescribeAttackStatistics Shield:DescribedrtAccess perisai: DescribeEmergencyContactSettings perisai: DescribeProtection perisai: DescribeProtectionGroup perisai: DescribeSubscription perisai: DisableApplicationLayerAutomaticResponse perisai: DisableProactiveEngagement perisai: DisassociateDRTLogBucket Perisai: DisassociateDrTrole perisai: DisassociateHealthCheck

Awalan layanan	Tindakan
	perisai: EnableApplicationLayerAutomaticResponse perisai: EnableProactiveEngagement perisai: GetSubscriptionState perisai: ListAttacks perisai: ListProtectionGroups perisai: ListProtections perisai: ListResourcesInProtectionGroup perisai: UpdateApplicationLayerAutomaticResponse perisai: UpdateEmergencyContactSettings perisai: UpdateProtectionGroup perisai: UpdateSubscription

Awalan layanan	Tindakan
penandatangan	penandatangan: AddProfilePermission
	penandatangan: CancelSigningProfile
	penandatangan: DescribeSigningJob
	penandatangan: GetSigningPlatform
	penandatangan: GetSigningProfile
	penandatangan: ListProfilePermissions
	penandatangan: ListSigningJobs
	penandatangan: ListSigningPlatforms
	penandatangan: ListSigningProfiles
	penandatangan: PutSigningProfile
	penandatangan: RemoveProfilePermission
	penandatangan: RevokeSignature
	penandatangan: RevokeSigningProfile
	penandatangan: SignPayload
	penandatangan: StartSigningJob

Awalan layanan	Tindakan
simspaceweaver	simspaceweaver: CreateSnapshot simspaceweaver: DeleteApp simspaceweaver: DeleteSimulation simspaceweaver: DescribeApp simspaceweaver: DescribeSimulation simspaceweaver: ListApps simspaceweaver: ListSimulations simspaceweaver: StartApp simspaceweaver: StartClock simspaceweaver: StartSimulation simspaceweaver: StopApp simspaceweaver: StopClock simspaceweaver: StopSimulation

Awalan layanan	Tindakan
sms	sms: CreateApp
	sms: CreateReplicationJob
	sms: DeleteApp
	sms: DeleteAppLaunchConfiguration
	sms: DeleteAppReplicationConfiguration
	sms: DeleteAppValidationConfiguration
	sms: DeleteReplicationJob
	sms: DeleteServerCatalog
	sms: DisassociateConnector
	sms: GenerateChangeSet
	sms: GenerateTemplate
	sms: GetApp
	sms: GetAppLaunchConfiguration
	sms: GetAppReplicationConfiguration
	sms: GetAppValidationConfiguration
	sms: GetAppValidationOutput
	sms: GetConnectors
	sms: GetReplicationJobs
	sms: GetReplicationRuns
	sms: GetServers
	sms: ImportAppCatalog

Awalan layanan	Tindakan
	sms: ImportServerCatalog
	sms: LaunchApp
	sms: ListApps
	sms: NotifyAppValidationOutput
	sms: PutAppLaunchConfiguration
	sms: PutAppReplicationConfiguration
	sms: PutAppValidationConfiguration
	sms: StartAppReplication
	sms: StartOnDemandAppReplication
	sms: StartOnDemandReplicationRun
	sms: StopAppReplication
	sms: TerminateApp
	sms: UpdateApp
	sms: UpdateReplicationJob

Awalan layanan	Tindakan
sms-suara	sms-suara: AssociateProtectConfiguration sms-suara: CreateConfigurationSet sms-suara: CreateConfigurationSetEventDestination sms-suara: CreateEventDestination sms-suara: CreateOptOutList sms-suara: CreatePool sms-suara: CreateProtectConfiguration sms-suara: CreateRegistration sms-suara: CreateRegistrationAssociation sms-suara: CreateRegistrationAttachment sms-suara: CreateRegistrationVersion sms-suara: CreateVerifiedDestinationNumber sms-suara: DeleteAccountDefaultProtectConfiguration sms-suara: DeleteConfigurationSet sms-suara: DeleteConfigurationSetEventDestination sms-suara: DeleteDefaultMessageType sms-suara: DeleteDefaultSenderId sms-suara: DeleteEventDestination sms-suara: DeleteKeyword sms-suara: DeleteMediaMessageSpendLimitOverride sms-suara: DeleteOptOutList

Awalan layanan	Tindakan
	sms-suara: DeleteOptedOutNumber sms-suara: DeletePool sms-suara: DeleteProtectConfiguration sms-suara: DeleteProtectConfigurationRuleSetNumberOverride sms-suara: DeleteRegistration sms-suara: DeleteRegistrationAttachment sms-suara: DeleteResourcePolicy sms-suara: DeleteTextMessageSpendLimitOverride sms-suara: DeleteVerifiedDestinationNumber sms-suara: DeleteVoiceMessageSpendLimitOverride sms-suara: DescribeAccountAttributes sms-suara: DescribeAccountLimits sms-suara: DescribeConfigurationSets sms-suara: DescribeKeywords sms-suara: DescribeOptOutLists sms-suara: DescribeOptedOutNumbers sms-suara: DescribePhoneNumbers sms-suara: DescribePools sms-suara: DescribeProtectConfigurations sms-suara: DescribeRegistrationAttachments sms-suara: DescribeRegistrationFieldDefinitions

Awalan layanan	Tindakan
	sms-suara: DescribeRegistrationFieldValues sms-suara: DescribeRegistrationSectionDefinitions sms-suara: DescribeRegistrationTypeDefinitions sms-suara: DescribeRegistrationVersions sms-suara: DescribeRegistrations sms-suara: DescribeSenderIds sms-suara: DescribeSpendLimits sms-suara: DescribeVerifiedDestinationNumbers sms-suara: DisassociateOriginationIdentity sms-suara: DisassociateProtectConfiguration sms-suara: DiscardRegistrationVersion sms-suara: GetConfigurationSetEventDestinations sms-suara: GetProtectConfigurationCountryRuleSet sms-suara: GetResourcePolicy sms-suara: ListConfigurationSets sms-suara: ListPoolOriginationIdentities sms-suara: ListProtectConfigurationRuleSetNumberOverrides sms-suara: ListRegistrationAssociations sms-suara: PutKeyword sms-suara: PutOptedOutNumber sms-suara: PutProtectConfigurationRuleSetNumberOverride

Awalan layanan	Tindakan
	sms-suara: PutResourcePolicy sms-suara: ReleasePhoneNumber sms-suara: ReleaseSenderId sms-suara: RequestPhoneNumber sms-suara: RequestSenderId sms-suara: SendDestinationNumberVerificationCode sms-suara: SetAccountDefaultProtectConfiguration sms-suara: SetDefaultMessageFeedbackEnabled sms-suara: SetDefaultMessageType sms-suara: SetDefaultSenderId sms-suara: SetMediaMessageSpendLimitOverride sms-suara: SetTextMessageSpendLimitOverride sms-suara: SetVoiceMessageSpendLimitOverride sms-suara: SubmitRegistrationVersion sms-suara: UpdateConfigurationSetEventDestination sms-suara: UpdateEventDestination sms-suara: UpdatePhoneNumber sms-suara: UpdatePool sms-suara: UpdateProtectConfiguration sms-suara: UpdateProtectConfigurationCountryRuleSet sms-suara: UpdateSenderId

Awalan layanan	Tindakan
bola salju	bola salju: CancelCluster
	bola salju: CancelJob
	bola salju: CreateAddress
	bola salju: CreateCluster
	bola salju: CreateJob
	bola salju: CreateLongTermPricing
	bola salju: CreateReturnShippingLabel
	bola salju: DescribeAddress
	bola salju: DescribeAddresses
	bola salju: DescribeCluster
	bola salju: DescribeJob
	bola salju: DescribeReturnShippingLabel
	bola salju: GetJobManifest
	bola salju: GetJobUnlockCode
	bola salju: GetSnowballUsage
	bola salju: GetSoftwareUpdates
	bola salju: ListClusterJobs
	bola salju: ListClusters
	bola salju: ListCompatibleImages
	bola salju: ListJobs
	bola salju: ListLongTermPricing

Awalan layanan	Tindakan
	bola salju: ListPickupLocations bola salju: ListServiceVersions bola salju: UpdateCluster bola salju: UpdateJob bola salju: UpdateJobShipmentState bola salju: UpdateLongTermPricing
sqs	persegi: AddPermission persegi: CancelMessageMoveTask persegi: CreateQueue persegi: DeleteQueue persegi: PurgeQueue persegi: RemovePermission persegi: SetQueueAttributes

Awalan layanan	Tindakan
ssm	ssm: AssociateOpsItemRelatedItem
	ssm: CancelCommand
	ssm: CancelMaintenanceWindowExecution
	ssm: CreateActivation
	ssm: CreateAssociation
	ssm: CreateAssociationBatch
	ssm: CreateDocument
	ssm: CreateMaintenanceWindow
	ssm: CreateOpsItem
	ssm: CreateOpsMetadata
	ssm: CreatePatchBaseline
	ssm: CreateResourceDataSync
	ssm: DeleteActivation
	ssm: DeleteAssociation
	ssm: DeleteCloudConnector
	ssm: DeleteDocument
	ssm: DeleteInventory
	ssm: DeleteMaintenanceWindow
	ssm: DeleteOpsItem
	ssm: DeleteOpsMetadata
	ssm: DeleteParameter

Awalan layanan	Tindakan
	ssm: DeleteParameters
	ssm: DeletePatchBaseline
	ssm: DeleteResourceDataSync
	ssm: DeleteResourcePolicy
	ssm: DeregisterManagedInstance
	ssm: DeregisterPatchBaselineForPatchGroup
	ssm: DeregisterTargetFromMaintenanceWindow
	ssm: DeregisterTaskFromMaintenanceWindow
	ssm: DescribeActivations
	ssm: DescribeAssociation
	ssm: DescribeAssociationExecutionTargets
	ssm: DescribeAssociationExecutions
	ssm: DescribeAutomationExecutions
	ssm: DescribeAutomationStepExecutions
	ssm: DescribeAvailablePatches
	ssm: DescribeDocument
	ssm: DescribeDocumentParameters
	ssm: DescribeDocumentPermission
	ssm: DescribeEffectiveInstanceAssociations
	ssm: DescribeEffectivePatchesForPatchBaseline
	ssm: DescribeInstanceAssociationsStatus

Awalan layanan	Tindakan
	ssm: DescribeInstanceInformation
	ssm: DescribeInstancePatchStates
	ssm: DescribeInstancePatchStatesForPatchGroup
	ssm: DescribeInstancePatches
	ssm: DescribeInstanceProperties
	ssm: DescribeInventoryDeletions
	ssm: DescribeMaintenanceWindowExecutionTaskInvocations
	ssm: DescribeMaintenanceWindowExecutionTasks
	ssm: DescribeMaintenanceWindowExecutions
	ssm: DescribeMaintenanceWindowSchedule
	ssm: DescribeMaintenanceWindowTargets
	ssm: DescribeMaintenanceWindowTasks
	ssm: DescribeMaintenanceWindows
	ssm: DescribeMaintenanceWindowsForTarget
	ssm: DescribeOpsItems
	ssm: DescribeParameters
	ssm: DescribePatchBaselines
	ssm: DescribePatchGroupState
	ssm: DescribePatchGroups
	ssm: DescribePatchProperties
	ssm: DescribeSessions

Awalan layanan	Tindakan
	ssm: DisassociateOpsItemRelatedItem
	ssm: GetAccessToken
	ssm: GetAutomationExecution
	ssm: GetCalendarState
	ssm: GetCloudConnector
	ssm: GetCommandInvocation
	ssm: GetConnectionStatus
	ssm: GetDefaultPatchBaseline
	ssm: GetDeployablePatchSnapshotForInstance
	ssm: GetDocument
	ssm: GetExecutionPreview
	ssm: GetInventory
	ssm: GetInventorySchema
	ssm: GetMaintenanceWindow
	ssm: GetMaintenanceWindowExecution
	ssm: GetMaintenanceWindowExecutionTask
	ssm: GetMaintenanceWindowExecutionTaskInvocation
	ssm: GetMaintenanceWindowTask
	ssm: GetOpsItem
	ssm: GetOpsMetadata
	ssm: GetOpsSummary

Awalan layanan	Tindakan
	ssm: GetParameter
	ssm: GetParameterHistory
	ssm: GetParameters
	ssm: GetParametersByPath
	ssm: GetPatchBaseline
	ssm: GetPatchBaselineForPatchGroup
	ssm: GetResourcePolicies
	ssm: GetServiceSetting
	ssm: LabelParameterVersion
	ssm: ListAssociationVersions
	ssm: ListAssociations
	ssm: ListCloudConnectors
	ssm: ListCommandInvocations
	ssm: ListCommands
	ssm: ListComplianceItems
	ssm: ListComplianceSummaries
	ssm: ListDocumentMetadataHistory
	ssm: ListDocumentVersions
	ssm: ListDocuments
	ssm: ListInstanceAssociations
	ssm: ListInventoryEntries

Awalan layanan	Tindakan
	ssm: ListNodes
	ssm: ListNodesSummary
	ssm: ListOpsItemEvents
	ssm: ListOpsItemRelatedItems
	ssm: ListOpsMetadata
	ssm: ListResourceComplianceSummaries
	ssm: ListResourceDataSync
	ssm: ModifyDocumentPermission
	ssm: PutComplianceItems
	ssm: PutInventory
	ssm: PutParameter
	ssm: PutResourcePolicy
	ssm: RegisterDefaultPatchBaseline
	ssm: RegisterManagedInstance
	ssm: RegisterPatchBaselineForPatchGroup
	ssm: RegisterTargetWithMaintenanceWindow
	ssm: RegisterTaskWithMaintenanceWindow
	ssm: ResetServiceSetting
	ssm: ResumeSession
	ssm: SendAutomationSignal
	ssm: SendCommand

Awalan layanan	Tindakan
	ssm: StartAssociationsOnce
	ssm: StartAutomationExecution
	ssm: StartChangeRequestExecution
	ssm: StartSession
	ssm: StopAutomationExecution
	ssm: TerminateSession
	ssm: UnlabelParameterVersion
	ssm: UpdateAssociation
	ssm: UpdateAssociationStatus
	ssm: UpdateCloudConnector
	ssm: UpdateDocument
	ssm: UpdateDocumentDefaultVersion
	ssm: UpdateDocumentMetadata
	ssm: UpdateInstanceInformation
	ssm: UpdateMaintenanceWindow
	ssm: UpdateMaintenanceWindowTarget
	ssm: UpdateMaintenanceWindowTask
	ssm: UpdateManagedInstanceRole
	ssm: UpdateOpsItem
	ssm: UpdateOpsMetadata
	ssm: UpdatePatchBaseline

Awalan layanan	Tindakan
	ssm: UpdateResourceDataSync ssm: UpdateServiceSetting ssm: ValidateCloudConnector

Awalan layanan	Tindakan
insiden ssm	insiden ssm-: BatchGetIncidentFindings insiden ssm-: CreateReplicationSet insiden ssm-: CreateResponsePlan insiden ssm-: CreateTimelineEvent insiden ssm-: DeleteIncidentRecord insiden ssm-: DeleteReplicationSet insiden ssm-: DeleteResourcePolicy insiden ssm-: DeleteResponsePlan insiden ssm-: DeleteTimelineEvent insiden ssm-: GetIncidentRecord insiden ssm-: GetReplicationSet insiden ssm-: GetResourcePolicies insiden ssm-: GetResponsePlan insiden ssm-: GetTimelineEvent insiden ssm-: ListIncidentFindings insiden ssm-: ListIncidentRecords insiden ssm-: ListRelatedItems insiden ssm-: ListReplicationSets insiden ssm-: ListResponsePlans insiden ssm-: ListTimelineEvents insiden ssm-: PutResourcePolicy

Awalan layanan	Tindakan
	insiden ssm-: StartIncident insiden ssm-: UpdateDeletionProtection insiden ssm-: UpdateIncidentRecord insiden ssm-: UpdateRelatedItems insiden ssm-: UpdateReplicationSet insiden ssm-: UpdateResponsePlan insiden ssm-: UpdateTimelineEvent

Awalan layanan	Tindakan
ssm-sap	ssm-sap: BackupDatabase
	ssm-sap: DeleteResourcePermission
	ssm-sap: DeregisterApplication
	ssm-sap: GetApplication
	ssm-sap: GetComponent
	ssm-sap: GetConfigurationCheckOperation
	ssm-sap: GetDatabase
	ssm-sap: GetOperation
	ssm-sap: GetResourcePermission
	ssm-sap: ListApplications
	ssm-sap: ListComponents
	ssm-sap: ListConfigurationCheckDefinitions
	ssm-sap: ListConfigurationCheckOperations
	ssm-sap: ListDatabases
	ssm-sap: ListOperationEvents
	ssm-sap: ListOperations
	ssm-sap: ListSubCheckResults
	ssm-sap: ListSubCheckRuleResults
	ssm-sap: PutResourcePermission
	ssm-sap: RegisterApplication
	ssm-sap: RestoreDatabase

Awalan layanan	Tindakan
	ssm-sap: StartApplication
	ssm-sap: StartApplicationRefresh
	ssm-sap: StartConfigurationChecks
	ssm-sap: StopApplication
	ssm-sap: UpdateApplicationSettings
	ssm-sap: UpdateHANABackupSettings

Awalan layanan	Tindakan
negara	negara: CreateActivity negara: CreateStateMachine negara: CreateStateMachineAlias negara: DeleteActivity negara: DeleteStateMachine negara: DeleteStateMachineAlias negara: DeleteStateMachineVersion negara: DescribeActivity negara: DescribeExecution negara: DescribeMapRun negara: DescribeStateMachine negara: DescribeStateMachineAlias negara: DescribeStateMachineForExecution negara: GetExecutionHistory negara: ListActivities negara: ListExecutions negara: ListMapRuns negara: ListStateMachineAliases negara: ListStateMachineVersions negara: ListStateMachines negara: SendTaskFailure

Awalan layanan	Tindakan
	negara: SendTaskHeartbeat negara bagian: SendTaskSuccess negara bagian: StartExecution negara bagian: StopExecution negara bagian: UpdateMapRun negara bagian: UpdateStateMachine negara bagian: UpdateStateMachineAlias negara bagian: ValidateStateMachineDefinition
sts	sts: AssumeRole sts: AssumeRoleWithSAML sts: AssumeRoleWithWebIdentity sts: DecodeAuthorizationMessage sts: GetAccessKeyInfo sts: GetCallerIdentity sts: GetFederationToken sts: GetSessionToken sts: GetWebIdentityToken

Awalan layanan	Tindakan
swf	swf: DeleteActivityType swf: DeleteWorkflowType swf: DeprecateActivityType swf: DeprecateDomain swf: DeprecateWorkflowType swf: DescribeActivityType swf: DescribeDomain swf: DescribeWorkflowType swf: ListActivityTypes swf: ListDomains swf: ListWorkflowTypes swf: RegisterActivityType swf: RegisterDomain swf: RegisterWorkflowType swf: UndeprecateActivityType swf: UndeprecateDomain swf: UndeprecateWorkflowType

Awalan layanan	Tindakan
sintetis	sintetis: AssociateResource
	sintetis: CreateCanary
	sintetis: CreateGroup
	sintetis: DeleteCanary
	sintetis: DeleteGroup
	sintetis: DescribeCanaries
	sintetis: DescribeCanariesLastRun
	sintetis: DescribeRuntimeVersions
	sintetis: DisassociateResource
	sintetis: GetCanary
	sintetis: GetCanaryRuns
	sintetis: GetGroup
	sintetis: ListAssociatedGroups
	sintetis: ListGroupResources
	sintetis: ListGroups
	sintetis: StartCanary
	sintetis: StartCanaryDryRun
	sintetis: StopCanary
	sintetis: UpdateCanary

Awalan layanan	Tindakan
tanda	Tag: DescribeReportCreation Tag: GetComplianceSummary Tag: GetResources Tag: StartReportCreation

Awalan layanan	Tindakan
textract	teks: AnalyzeDocument
	teks: AnalyzeExpense
	Textract:AnalyzeID
	teks: CreateAdapter
	teks: CreateAdapterVersion
	teks: DeleteAdapter
	teks: DeleteAdapterVersion
	teks: DetectDocumentText
	teks: GetAdapter
	teks: GetAdapterVersion
	teks: GetDocumentAnalysis
	teks: GetDocumentTextDetection
	teks: GetExpenseAnalysis
	teks: GetLendingAnalysis
	teks: GetLendingAnalysisSummary
	teks: ListAdapterVersions
	teks: ListAdapters
	teks: StartDocumentAnalysis
	teks: StartDocumentTextDetection
	teks: StartExpenseAnalysis
	teks: StartLendingAnalysis

Awalan layanan	Tindakan
	teks: UpdateAdapter

Awalan layanan	Tindakan
aliran waktu	aliran waktu: CancelQuery aliran waktu: CreateDatabase aliran waktu: CreateScheduledQuery aliran waktu: CreateTable aliran waktu: DeleteDatabase aliran waktu: DeleteScheduledQuery aliran waktu: DeleteTable aliran waktu: DescribeAccountSettings aliran waktu: DescribeDatabase aliran waktu: DescribeScheduledQuery aliran waktu: DescribeTable aliran waktu: ExecuteScheduledQuery aliran waktu: ListBatchLoadTasks aliran waktu: ListDatabases aliran waktu: ListScheduledQueries aliran waktu: ListTables aliran waktu: PrepareQuery aliran waktu: UpdateAccountSettings aliran waktu: UpdateDatabase aliran waktu: UpdateScheduledQuery aliran waktu: UpdateTable

Awalan layanan	Tindakan
tnb	tnb: CancelSolNetworkOperation
	tnb: CreateSolFunctionPackage
	tnb: CreateSolNetworkInstance
	tnb: CreateSolNetworkPackage
	tnb: DeleteSolFunctionPackage
	tnb: DeleteSolNetworkInstance
	tnb: DeleteSolNetworkPackage
	tnb: GetSolFunctionInstance
	tnb: GetSolFunctionPackage
	tnb: GetSolFunctionPackageContent
	tnb: GetSolFunctionPackageDescriptor
	tnb: GetSolNetworkInstance
	tnb: GetSolNetworkOperation
	tnb: GetSolNetworkPackage
	tnb: GetSolNetworkPackageContent
	tnb: GetSolNetworkPackageDescriptor
	tnb: InstantiateSolNetworkInstance
	tnb: ListSolFunctionInstances
	tnb: ListSolFunctionPackages
	tnb: ListSolNetworkInstances
	tnb: ListSolNetworkOperations

Awalan layanan	Tindakan
	tnb: ListSolNetworkPackages
	tnb: PutSolFunctionPackageContent
	tnb: PutSolNetworkPackageContent
	tnb: TerminateSolNetworkInstance
	tnb: UpdateSolFunctionPackage
	tnb: UpdateSolNetworkInstance
	tnb: UpdateSolNetworkPackage
	tnb: ValidateSolFunctionPackageContent
	tnb: ValidateSolNetworkPackageContent

Awalan layanan	Tindakan
mentranskripsikan	transkripsikan: CreateCallAnalyticsCategory transkripsikan: CreateLanguageModel transkripsikan: CreateMedicalVocabulary transkripsikan: CreateVocabulary transkripsikan: CreateVocabularyFilter transkripsikan: DeleteCallAnalyticsCategory transkripsikan: DeleteCallAnalyticsJob transkripsikan: DeleteLanguageModel transkripsikan: DeleteMedicalScribeJob transkripsikan: DeleteMedicalTranscriptionJob transkripsikan: DeleteMedicalVocabulary transkripsikan: DeleteTranscriptionJob transkripsikan: DeleteVocabulary transkripsikan: DeleteVocabularyFilter transkripsikan: DescribeLanguageModel transkripsikan: GetCallAnalyticsCategory transkripsikan: GetCallAnalyticsJob transkripsikan: GetMedicalScribeJob transkripsikan: GetMedicalTranscriptionJob transkripsikan: GetMedicalVocabulary transkripsikan: GetTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: GetVocabulary transkripsikan: GetVocabularyFilter transkripsikan: ListCallAnalyticsCategories transkripsikan: ListCallAnalyticsJobs transkripsikan: ListLanguageModels transkripsikan: ListMedicalScribeJobs transkripsikan: ListMedicalTranscriptionJobs transkripsikan: ListMedicalVocabularies transkripsikan: ListTranscriptionJobs transkripsikan: ListVocabularies transkripsikan: ListVocabularyFilters transkripsikan: StartCallAnalyticsJob transkripsikan: StartCallAnalyticsStreamTranscription transkripsikan: StartCallAnalyticsStreamTranscriptionWebSocket transkripsikan: StartMedicalScribeJob transkripsikan: StartMedicalStreamTranscription transkripsikan: StartMedicalStreamTranscriptionWebSocket transkripsikan: StartMedicalTranscriptionJob transkripsikan: StartStreamTranscription transkripsikan: StartStreamTranscriptionWebSocket transkripsikan: StartTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: UpdateCallAnalyticsCategory transkripsikan: UpdateMedicalVocabulary transkripsikan: UpdateVocabulary transkripsikan: UpdateVocabularyFilter

Awalan layanan	Tindakan
transfer	transfer: CreateAccess
	transfer: CreateAgreement
	transfer: CreateConnector
	transfer: CreateProfile
	transfer: CreateServer
	transfer: CreateUser
	transfer: CreateWebApp
	transfer: CreateWorkflow
	transfer: DeleteAccess
	transfer: DeleteAgreement
	transfer: DeleteCertificate
	transfer: DeleteConnector
	transfer: DeleteHostKey
	transfer: DeleteProfile
	transfer: DeleteServer
	transfer: DeleteSshPublicKey
	transfer: DeleteUser
	transfer: DeleteWebApp
	transfer: DeleteWebAppCustomization
	transfer: DeleteWorkflow
	transfer: DescribeAccess

Awalan layanan	Tindakan
	transfer: DescribeAgreement
	transfer: DescribeCertificate
	transfer: DescribeConnector
	transfer: DescribeExecution
	transfer: DescribeHostKey
	transfer: DescribeProfile
	transfer: DescribeSecurityPolicy
	transfer: DescribeServer
	transfer: DescribeUser
	transfer: DescribeWebApp
	transfer: DescribeWebAppCustomization
	transfer: DescribeWorkflow
	transfer: ImportCertificate
	transfer: ImportHostKey
	transfer: ImportSshPublicKey
	transfer: ListAccesses
	transfer: ListCertificates
	transfer: ListConnectors
	transfer: ListExecutions
	transfer: ListFileTransferResults
	transfer: ListHostKeys

Awalan layanan	Tindakan
	transfer: ListProfiles
	transfer: ListSecurityPolicies
	transfer: ListServers
	transfer: ListUsers
	transfer: ListWebApps
	transfer: ListWorkflows
	transfer: SendWorkflowStepState
	transfer: StartDirectoryListing
	transfer: StartFileTransfer
	transfer: StartRemoteDelete
	transfer: StartRemoteMove
	transfer: StartServer
	transfer: StopServer
	transfer: TestConnection
	transfer: TestIdentityProvider
	transfer: UpdateAccess
	transfer: UpdateAgreement
	transfer: UpdateCertificate
	transfer: UpdateConnector
	transfer: UpdateHostKey
	transfer: UpdateProfile

Awalan layanan	Tindakan
	transfer: UpdateServer transfer: UpdateUser transfer: UpdateWebApp transfer: UpdateWebAppCustomization
menerjemahkan	menerjemahkan: CreateParallelData menerjemahkan: DeleteParallelData menerjemahkan: DeleteTerminology menerjemahkan: DescribeTextTranslationJob menerjemahkan: GetParallelData menerjemahkan: GetTerminology menerjemahkan: ImportTerminology menerjemahkan: ListLanguages menerjemahkan: ListParallelData menerjemahkan: ListTerminologies menerjemahkan: ListTextTranslationJobs menerjemahkan: StartTextTranslationJob menerjemahkan: StopTextTranslationJob menerjemahkan: TranslateDocument menerjemahkan: TranslateText menerjemahkan: UpdateParallelData

Awalan layanan	Tindakan
voiceid	voiceid: AssociateFraudster
	voiceid: CreateDomain
	voiceid: CreateWatchlist
	voiceid: DeleteDomain
	voiceid: DeleteFraudster
	voiceid: DeleteSpeaker
	voiceid: DeleteWatchlist
	voiceid: DescribeDomain
	voiceid: DescribeFraudster
	voiceid: DescribeFraudsterRegistrationJob
	voiceid: DescribeSpeaker
	voiceid: DescribeSpeakerEnrollmentJob
	voiceid: DescribeWatchlist
	voiceid: DisassociateFraudster
	voiceid: EvaluateSession
	voiceid: ListDomains
	voiceid: ListFraudsterRegistrationJobs
	voiceid: ListFraudsters
	voiceid: ListSpeakerEnrollmentJobs
	voiceid: ListSpeakers
	voiceid: ListWatchlists

Awalan layanan	Tindakan
	voiceid: OptOutSpeaker voiceid: StartFraudsterRegistrationJob voiceid: StartSpeakerEnrollmentJob voiceid: UpdateDomain voiceid: UpdateWatchlist

Awalan layanan	Tindakan
vpc-kisi	vpc-kisi: CreateAccessLogSubscription vpc-kisi: CreateListener vpc-kisi: CreateResourceConfiguration vpc-kisi: CreateResourceGateway vpc-kisi: CreateRule vpc-kisi: CreateService vpc-kisi: CreateServiceNetwork vpc-kisi: CreateServiceNetworkResourceAssociation vpc-kisi: CreateServiceNetworkServiceAssociation vpc-kisi: CreateServiceNetworkVpcAssociation vpc-kisi: CreateTargetGroup vpc-kisi: DeleteAccessLogSubscription vpc-kisi: DeleteAuthPolicy vpc-kisi: DeleteDomainVerification vpc-kisi: DeleteListener vpc-kisi: DeleteResourceConfiguration vpc-kisi: DeleteResourceEndpointAssociation vpc-kisi: DeleteResourceGateway vpc-kisi: DeleteResourcePolicy vpc-kisi: DeleteRule vpc-kisi: DeleteService

Awalan layanan	Tindakan
	vpc-kisi: DeleteServiceNetwork vpc-kisi: DeleteServiceNetworkResourceAssociation vpc-kisi: DeleteServiceNetworkServiceAssociation vpc-kisi: DeleteServiceNetworkVpcAssociation vpc-kisi: DeleteTargetGroup vpc-kisi: DeregisterTargets vpc-kisi: GetAccessLogSubscription vpc-kisi: GetAuthPolicy vpc-kisi: GetDomainVerification vpc-kisi: GetListener vpc-kisi: GetResourceConfiguration vpc-kisi: GetResourceGateway vpc-kisi: GetResourcePolicy vpc-kisi: GetRule vpc-kisi: GetService vpc-kisi: GetServiceNetwork vpc-kisi: GetServiceNetworkResourceAssociation vpc-kisi: GetServiceNetworkServiceAssociation vpc-kisi: GetServiceNetworkVpcAssociation vpc-kisi: GetTargetGroup vpc-kisi: ListAccessLogSubscriptions

Awalan layanan	Tindakan
	vpc-kisi: ListDomainVerifications vpc-kisi: ListListeners vpc-kisi: ListResourceConfigurations vpc-kisi: ListResourceEndpointAssociations vpc-kisi: ListResourceGateways vpc-kisi: ListRules vpc-kisi: ListServiceNetworkResourceAssociations vpc-kisi: ListServiceNetworkServiceAssociations vpc-kisi: ListServiceNetworkVpcAssociations vpc-kisi: ListServiceNetworkVpcEndpointAssociations vpc-kisi: ListServiceNetworks vpc-kisi: ListServices vpc-kisi: ListTargetGroups vpc-kisi: ListTargets vpc-kisi: PutAuthPolicy vpc-kisi: PutResourcePolicy vpc-kisi: RegisterTargets vpc-kisi: StartDomainVerification vpc-kisi: UpdateAccessLogSubscription vpc-kisi: UpdateListener vpc-kisi: UpdateResourceConfiguration

Awalan layanan	Tindakan
	vpc-kisi: UpdateResourceGateway vpc-kisi: UpdateRule vpc-kisi: UpdateService vpc-kisi: UpdateServiceNetwork vpc-kisi: UpdateServiceNetworkVpcAssociation vpc-kisi: UpdateTargetGroup

Awalan layanan	Tindakan
wafv2	wafv2: ACL AssociateWeb
	wafv2: CheckCapacity
	WAFv2:createApiKey
	WAFv2: CreateIPSet
	wafv2: CreateRegexPatternSet
	wafv2: CreateRuleGroup
	wafv2: ACL CreateWeb
	WAFv2: DeleteApiKey
	wafv2: DeleteFirewallManagerRuleGroups
	WAFv2: DeleteIPSet
	wafv2: DeleteLoggingConfiguration
	wafv2: DeletePermissionPolicy
	wafv2: DeleteRegexPatternSet
	wafv2: DeleteRuleGroup
	wafv2: ACL DeleteWeb
	wafv2: DescribeAllManagedProducts
	wafv2: DescribeManagedProductsByVendor
	wafv2: DescribeManagedRuleGroup
	wafv2: ACL DisassociateWeb
	wafv2: GenerateMobileSdkReleaseUrl
	wafv2: GetDecryptedAPIKey

Awalan layanan	Tindakan
	WAFV2: GetIpSet
	wafv2: GetLoggingConfiguration
	wafv2: GetManagedRuleSet
	wafv2: GetMobileSdkRelease
	wafv2: GetRateBasedStatementManagedKeys
	wafv2: GetRegexPatternSet
	wafv2: GetRevenueStatistics
	wafv2: GetRevenueStatisticsSummary
	wafv2: GetRevenueStatisticsTimeSeries
	wafv2: GetRuleGroup
	wafv2: GetSampledRequests
	wafv2: GetWebACLForResource
	WAFv2:ListApiKeys
	wafv2: ListAvailableManagedRuleGroupVersions
	wafv2: ListAvailableManagedRuleGroups
	WAFv2: ListipSets
	wafv2: ListLoggingConfigurations
	wafv2: ListManagedRuleSets
	wafv2: ListMobileSdkReleases
	wafv2: ListRegexPatternSets
	wafv2: ListResourcesForWebACL

Awalan layanan	Tindakan
	wafv2: ListRuleGroups
	wafv2: ListSettlementRecords
	wafv2: ListWebACLs
	wafv2: PutLoggingConfiguration
	wafv2: PutManagedRuleSetVersions
	WAFv2: UpdateIPSet
	wafv2: UpdateManagedRuleSetVersionExpiryDate
	wafv2: UpdateRegexPatternSet
	wafv2: UpdateRuleGroup
	wafv2: ACL UpdateWeb

Awalan layanan	Tindakan
wellarchitected	wellarchitected: AssociateLenses wellarchitected: AssociateProfiles wellarchitected: CreateLensShare wellarchitected: CreateLensVersion wellarchitected: CreateMilestone wellarchitected: CreateProfile wellarchitected: CreateProfileShare wellarchitected: CreateReviewTemplate wellarchitected: CreateWorkload wellarchitected: CreateWorkloadShare wellarchitected: DeleteLens wellarchitected: DeleteLensShare wellarchitected: DeleteProfile wellarchitected: DeleteProfileShare wellarchitected: DeleteReviewTemplate wellarchitected: DeleteTemplateShare wellarchitected: DeleteWorkload wellarchitected: DeleteWorkloadShare wellarchitected: DisassociateLenses wellarchitected: DisassociateProfiles wellarchitected: ExportLens

Awalan layanan	Tindakan
	wellarchitected: GetAnswer wellarchitected: GetConsolidatedReport wellarchitected: GetGlobalSettings wellarchitected: GetLens wellarchitected: GetLensReview wellarchitected: GetLensReviewReport wellarchitected: GetLensVersionDifference wellarchitected: GetMilestone wellarchitected: GetProfile wellarchitected: GetProfileTemplate wellarchitected: GetReviewTemplate wellarchitected: GetReviewTemplateAnswer wellarchitected: GetReviewTemplateLensReview wellarchitected: GetWorkload wellarchitected: ImportLens wellarchitected: ListAnswers wellarchitected: ListCheckDetails wellarchitected: ListCheckSummaries wellarchitected: ListLensReviewImprovements wellarchitected: ListLensReviews wellarchitected: ListLensShares

Awalan layanan	Tindakan
	wellarchitected: ListLenses wellarchitected: ListMilestones wellarchitected: ListNotifications wellarchitected: ListProfileNotifications wellarchitected: ListProfileShares wellarchitected: ListProfiles wellarchitected: ListReviewTemplateAnswers wellarchitected: ListReviewTemplates wellarchitected: ListShareInvitations wellarchitected: ListTemplateShares wellarchitected: ListWorkloadShares wellarchitected: ListWorkloads wellarchitected: UpdateAnswer wellarchitected: UpdateGlobalSettings wellarchitected: UpdateIntegration wellarchitected: UpdateLensReview wellarchitected: UpdateProfile wellarchitected: UpdateReviewTemplate wellarchitected: UpdateReviewTemplateLensReview wellarchitected: UpdateShareInvitation wellarchitected: UpdateWorkload

Awalan layanan	Tindakan
	wellarchitected: UpdateWorkloadShare wellarchitected: UpgradeLensReview wellarchitected: UpgradeProfileVersion wellarchitected: UpgradeReviewTemplateLensReview

Awalan layanan	Tindakan
kebijaksanaan	kebijaksanaan: CreateAssistant
	kebijaksanaan: CreateAssistantAssociation
	kebijaksanaan: CreateContent
	kebijaksanaan: CreateKnowledgeBase
	kebijaksanaan: CreateQuickResponse
	kebijaksanaan: CreateSession
	kebijaksanaan: DeleteAssistant
	kebijaksanaan: DeleteAssistantAssociation
	kebijaksanaan: DeleteContent
	kebijaksanaan: DeleteImportJob
	kebijaksanaan: DeleteKnowledgeBase
	kebijaksanaan: DeleteQuickResponse
	kebijaksanaan: GetAssistant
	kebijaksanaan: GetAssistantAssociation
	kebijaksanaan: GetContent
	kebijaksanaan: GetContentAssociation
	kebijaksanaan: GetContentSummary
	kebijaksanaan: GetImportJob
	kebijaksanaan: GetKnowledgeBase
	kebijaksanaan: GetRecommendations
	kebijaksanaan: GetSession

Awalan layanan	Tindakan
	kebijaksanaan: ListAssistantAssociations
	kebijaksanaan: ListAssistants
	kebijaksanaan: ListContentAssociations
	kebijaksanaan: ListContents
	kebijaksanaan: ListImportJobs
	kebijaksanaan: ListKnowledgeBases
	kebijaksanaan: ListQuickResponses
	kebijaksanaan: NotifyRecommendationsReceived
	kebijaksanaan: QueryAssistant
	kebijaksanaan: RemoveKnowledgeBaseTemplateUri
	kebijaksanaan: SearchContent
	kebijaksanaan: SearchQuickResponses
	kebijaksanaan: SearchSessions
	kebijaksanaan: StartContentUpload
	kebijaksanaan: StartImportJob
	kebijaksanaan: UpdateContent
	kebijaksanaan: UpdateKnowledgeBaseTemplateUri
	kebijaksanaan: UpdateQuickResponse
	kebijaksanaan: UpdateSession

Awalan layanan	Tindakan
tautan kerja	tautan kerja: AssociateDomain tautan kerja: AssociateWebsiteAuthorizationProvider tautan kerja: AssociateWebsiteCertificateAuthority tautan kerja: CreateFleet tautan kerja: DeleteFleet tautan kerja: DescribeAuditStreamConfiguration tautan kerja: DescribeCompanyNetworkConfiguration tautan kerja: DescribeDevice tautan kerja: DescribeDevicePolicyConfiguration tautan kerja: DescribeDomain tautan kerja: DescribeFleetMetadata tautan kerja: DescribeIdentityProviderConfiguration tautan kerja: DescribeWebsiteCertificateAuthority tautan kerja: DisassociateDomain tautan kerja: DisassociateWebsiteAuthorizationProvider tautan kerja: DisassociateWebsiteCertificateAuthority tautan kerja: ListDevices tautan kerja: ListDomains tautan kerja: ListFleets tautan kerja: ListWebsiteAuthorizationProviders tautan kerja: ListWebsiteCertificateAuthorities

Awalan layanan	Tindakan
	tautan kerja: RestoreDomainAccess tautan kerja: RevokeDomainAccess tautan kerja: SignOutUser tautan kerja: UpdateAuditStreamConfiguration tautan kerja: UpdateCompanyNetworkConfiguration tautan kerja: UpdateDevicePolicyConfiguration tautan kerja: UpdateDomainMetadata tautan kerja: UpdateFleetMetadata tautan kerja: UpdateIdentityProviderConfiguration

Awalan layanan	Tindakan
ruang kerja	ruang kerja: AcceptAccountLinkInvitation
	ruang kerja: AssociateConnectionAlias
	ruang kerja: AssociateIpGroups
	ruang kerja: AssociateWorkspaceApplication
	ruang kerja: CopyWorkspaceImage
	ruang kerja: CreateAccountLinkInvitation
	ruang kerja: CreateConnectClientAddIn
	ruang kerja: CreateConnectionAlias
	ruang kerja: CreateIpGroup
	ruang kerja: CreateStandbyWorkspaces
	ruang kerja: CreateUpdatedWorkspaceImage
	ruang kerja: CreateWorkspaceBundle
	ruang kerja: CreateWorkspaceImage
	ruang kerja: CreateWorkspaces
	ruang kerja: CreateWorkspacesPool
	ruang kerja: DeleteAccountLinkInvitation
	ruang kerja: DeleteClientBranding
	ruang kerja: DeleteConnectClientAddIn
	ruang kerja: DeleteConnectionAlias
	ruang kerja: DeleteIpGroup
	ruang kerja: DeleteWorkspaceBundle

Awalan layanan	Tindakan
	ruang kerja: DeleteWorkspaceImage
	ruang kerja: DeployWorkspaceApplications
	ruang kerja: DeregisterWorkspaceDirectory
	ruang kerja: DescribeAccount
	ruang kerja: DescribeAccountModifications
	ruang kerja: DescribeApplicationAssociations
	ruang kerja: DescribeApplications
	ruang kerja: DescribeBundleAssociations
	ruang kerja: DescribeClientBranding
	ruang kerja: DescribeClientProperties
	ruang kerja: DescribeConnectClientAddIns
	ruang kerja: DescribeConnectionAliasPermissions
	ruang kerja: DescribeConnectionAliases
	ruang kerja: DescribeCustomWorkspaceImageImport
	ruang kerja: DescribeImageAssociations
	ruang kerja: DescribeIpGroups
	ruang kerja: DescribeWorkspaceAssociations
	ruang kerja: DescribeWorkspaceBundles
	ruang kerja: DescribeWorkspaceDirectories
	ruang kerja: DescribeWorkspaceImagePermissions
	ruang kerja: DescribeWorkspaceSnapshots

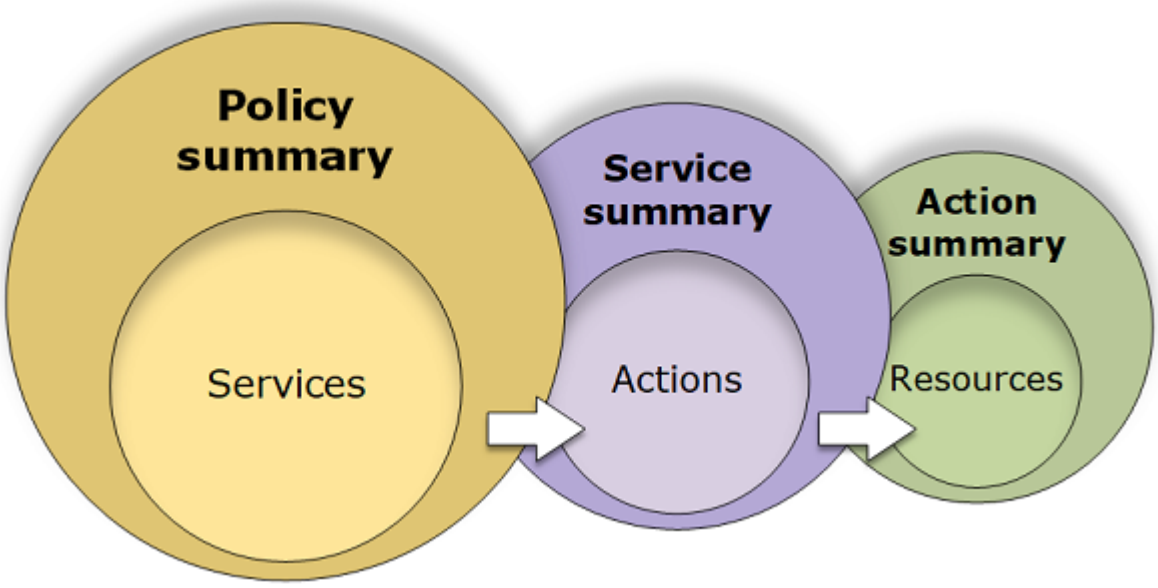
Awalan layanan	Tindakan
	ruang kerja: DescribeWorkspaces
	ruang kerja: DescribeWorkspacesConnectionStatus
	ruang kerja: DescribeWorkspacesPoolSessions
	ruang kerja: DescribeWorkspacesPools
	ruang kerja: DisassociateConnectionAlias
	ruang kerja: DisassociateIpGroups
	ruang kerja: DisassociateWorkspaceApplication
	ruang kerja: GetAccountLink
	ruang kerja: ImportClientBranding
	ruang kerja: ImportWorkspaceImage
	ruang kerja: ListAccountLinks
	ruang kerja: ListAvailableManagementCidrRanges
	ruang kerja: MigrateWorkspace
	ruang kerja: ModifyAccount
	ruang kerja: ModifyCertificateBasedAuthProperties
	ruang kerja: ModifyClientProperties
	ruang kerja: ModifyEndpointEncryptionMode
	ruang kerja: ModifySamlProperties
	ruang kerja: ModifySelfservicePermissions
	ruang kerja: ModifyStreamingProperties
	ruang kerja: ModifyWorkspaceAccessProperties

Awalan layanan	Tindakan
	ruang kerja: ModifyWorkspaceCreationProperties
	ruang kerja: ModifyWorkspaceProperties
	ruang kerja: ModifyWorkspaceState
	ruang kerja: RebootWorkspaces
	ruang kerja: RebuildWorkspaces
	ruang kerja: RegisterWorkspaceDirectory
	ruang kerja: RejectAccountLinkInvitation
	ruang kerja: RestoreWorkspace
	ruang kerja: StartWorkspaces
	ruang kerja: StartWorkspacesPool
	ruang kerja: StopWorkspaces
	ruang kerja: StopWorkspacesPool
	ruang kerja: TerminateWorkspaces
	ruang kerja: TerminateWorkspacesPool
	ruang kerja: TerminateWorkspacesPoolSession
	ruang kerja: UpdateConnectClientAddIn
	ruang kerja: UpdateConnectionAliasPermission
	ruang kerja: UpdateWorkspaceBundle
	ruang kerja: UpdateWorkspaceImagePermission
	ruang kerja: UpdateWorkspacesPool

Awalan layanan	Tindakan
xray	xray: CreateGroup xray: CreateSamplingRule xray: DeleteGroup xray: DeleteResourcePolicy xray: DeleteSamplingRule xray: GetEncryptionConfig xray: GetGroup xray: GetGroups xray: GetInsight xray: GetInsightEvents xray: GetInsightImpactGraph xray: GetInsightSummaries xray: GetSamplingRules xray: ListResourcePolicies xray: PutEncryptionConfig xray: PutResourcePolicy xray: UpdateGroup xray: UpdateSamplingRule

Ringkasan kebijakan

Konsol IAM mencakup tabel ringkasan kebijakan yang menjelaskan tingkat akses, sumber daya, dan kondisi yang diperbolehkan atau ditolak untuk setiap layanan dalam kebijakan. Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan kebijakan mencakup daftar layanan. Pilih layanan yang tertera untuk melihat ringkasan layanan. Tabel ringkasan ini mencakup daftar tindakan dan izin terkait untuk layanan yang dipilih. Anda dapat memilih tindakan dari tabel tersebut untuk melihat ringkasan tindakan. Tabel ini mencakup daftar sumber daya dan IT untuk tindakan yang dipilih.



Anda dapat melihat rangkuman kebijakan di halaman Pengguna atau Peran untuk semua kebijakan (terkelola dan inline) yang terlampir pada pengguna tersebut. Lihat rangkuman dalam halaman Kebijakan untuk seluruh kebijakan terkelola Kebijakan terkel AWS ola mencakup kebijakan terkel AWS ola, kebijakan fungsi pekerjaan terkelola, dan kebijakan yang dikelola pelanggan. Anda dapat melihat rangkuman untuk kebijakan-kebijakan ini di Kebijakan halaman tidak GApeduli apakah mereka terpasang pada seseorang atau IT IAM lainnya.

Anda dapat menggunakan informasi dalam rangkuman kebijakan untuk memahami izin yang diizinkan atau ditolak oleh kebijakan Anda. Ringkasan kebijakan dapat membantu Anda pemecahan masalah dan memperbaiki kebijakan yang tidak memberikan izin yang Anda harapkan.

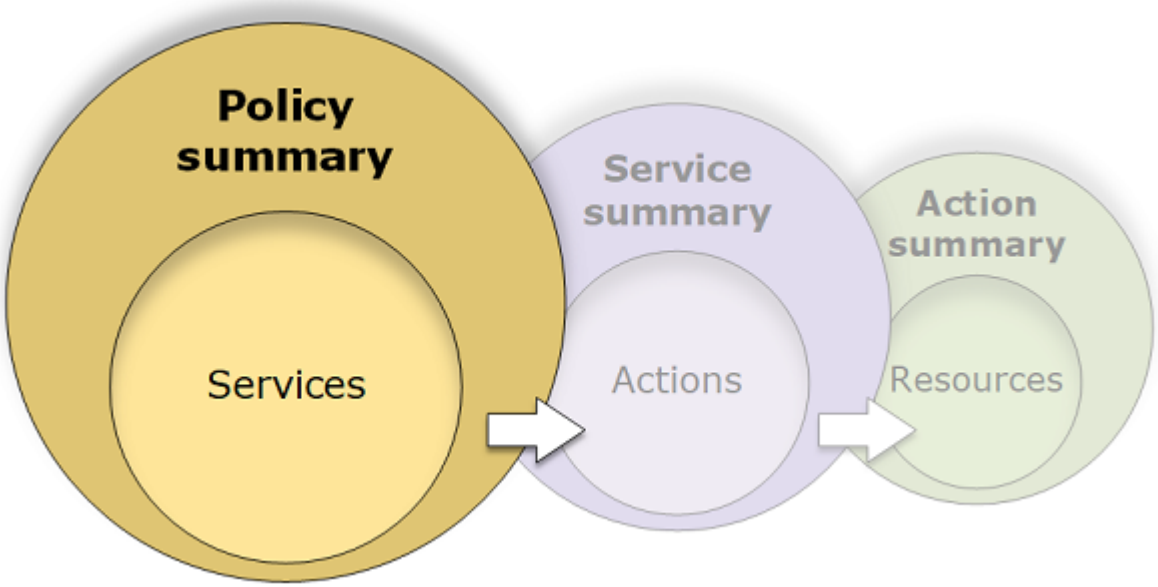
Untuk petunjuk langkah demi langkah tentang melihat ringkasan, lihatLihat ringkasan kebijakan,Lihat ringkasan layanan, danLihat ringkasan tindakan.

Topik

- Ringkasan kebijakan (daftar layanan)
- Tingkat akses dalam ringkasan kebijakan
- Ringkasan layanan (daftar tindakan)
- Ringkasan tindakan (daftar sumber daya)
- Contoh ringkasan kebijakan

Ringkasan kebijakan (daftar layanan)

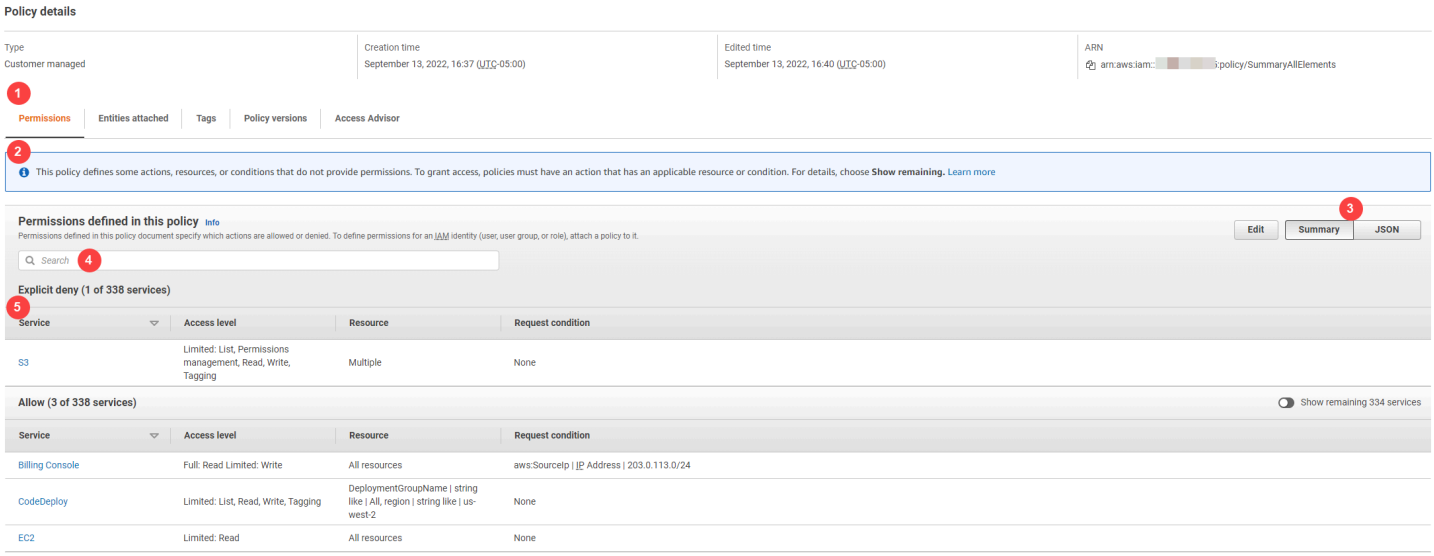
Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan kebijakan mencakup daftar layanan dan ringkasan izin yang ditentukan oleh kebijakan yang dipilih.



Tabel ringkasan kebijakan dikelompokkan menjadi satu atau lebih bagian Layanan yang tidak terkategorikan, Penolakan secara eksplisit , dan Izinkan. Jika kebijakan mencakup layanan yang tidak dikenali oleh IAM, layanan tersebut termasuk dalam Layanan yang tidak terkategorikan di tabel. Jika IAM mengenali layanan, maka ia termasuk dalam bagian tabel Penolakan secara eksplisit atau Izinkan tergantung pada efek kebijakan (`Deny` atau `Allow`).

Memahami elemen ringkasan kebijakan

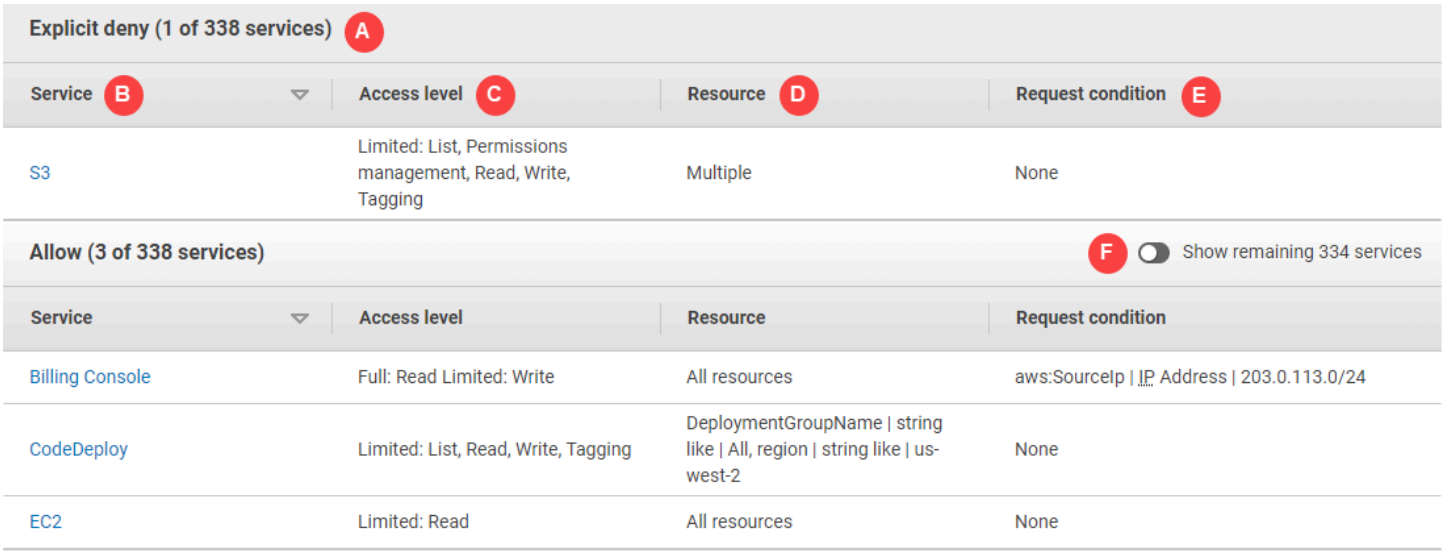
Dalam contoh halaman detail kebijakan berikut, kebijakan tersebut adalah SummaryAllElementskebijakan terkelola (kebijakan yang dikelola pelanggan) yang dilampirkan langsung ke pengguna. Kebijakan ini diperluas untuk menunjukkan ringkasan kebijakan.



Pada gambar sebelumnya, ringkasan kebijakan terlihat dari dalam halaman Kebijakan:

1. Tab Izin menyertakan izin yang ditentukan dalam kebijakan.
2. Jika kebijakan tidak memberikan izin untuk semua tindakan, sumber daya, dan syarat yang ditentukan dalam kebijakan, maka peringatan atau banner muncul di bagian atas halaman. Ringkasan kebijakan kemudian mencakup perincian tentang masalah. Untuk mempelajari bagaimana rangkuman kebijakan membantu Anda memahami dan mengatasi masalah izin yang diberikan oleh kebijakan Anda, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.
3. Gunakan tombol Ringkasan dan JSON untuk beralih antara ringkasan kebijakan dan dokumen kebijakan JSON.
4. Gunakan kotak Pencarian untuk mengurangi daftar layanan dan menemukan layanan tertentu.
5. Tampilan yang diperluas menampilkan detail tambahan kebijakan SummaryAllElements.

Gambar tabel ringkasan kebijakan berikut menunjukkan SummaryAllElementskebijakan yang diperluas pada halaman detail kebijakan.



Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

Pada gambar sebelumnya, ringkasan kebijakan terlihat dari dalam halaman Kebijakan:

A. Untuk layanan yang diakui IAM, layanan diatur sesuai dengan apakah kebijakan mengizinkan atau secara eksplisit menghalangi penggunaan layanan. Dalam contoh ini, kebijakan tersebut mencakup `Deny` pernyataan untuk layanan Amazon S3 dan `Allow` pernyataan untuk layanan Penagihan, CodeDeploy, dan Amazon EC2.

B. Layanan – Kolom ini mencantumkan layanan yang ditegaskan dalam kebijakan dan menyajikan detail untuk setiap layanan. Setiap nama layanan dalam tabel ringkasan kebijakan adalah tautan ke ringkasan layanan yang dijelaskan dalam Ringkasan layanan (daftar tindakan). Dalam contoh ini, izin ditentukan untuk layanan Amazon S3, Penagihan CodeDeploy, dan Amazon EC2.

C. Tingkat akses — Kolom ini memberi tahu apakah tindakan di setiap tingkat akses (`List`,`Read`,`Write`,`Permission Management`,, dan`Tagging`) memiliki `Full` atau `Limited` izin yang ditentukan dalam kebijakan. Untuk detail dan contoh tambahan dari ringkasan tingkat akses, lihat Tingkat akses dalam ringkasan kebijakan.

- Akses penuh – Entri ini menunjukkan bahwa layanan memiliki akses ke semua tindakan, di dalamnya tersedia empat tingkat akses untuk layanan tersebut.
- Jika entri tidak mencakup Akses penuh, maka layanan memiliki akses ke beberapa tetapi tidak semua tindakan untuk layanan tersebut. Akses kemudian didefinisikan dengan deskripsi berikut untuk masing-masing klasifikasi tingkat akses (`List`,,, `Read` `WritePermission Management`, dan`Tagging`):

  Lengkap: Kebijakan ini menyediakan akses ke semua tindakan yang di dalamnya mencantumkan setiap klasifikasi tingkat akses. Dalam contoh ini, kebijakan ini memberikan akses ke semua `Read` aksi nyata.

Terbatas: Kebijakan ini menyediakan akses ke satu atau beberapa tetapi tidak semua tindakan tercantum dalam klasifikasi tingkat akses. Dalam contoh ini, kebijakan ini memberikan akses ke beberapa `Write` aksi nyata.

D. Sumber Daya – Kolom ini menunjukkan sumber daya yang ditentukan kebijakan untuk setiap layanan.

- Banyak – Kebijakan ini mencakup lebih dari satu, tetapi tidak semua sumber daya di balik layanan tersebut. Dalam contoh ini, akses secara eksplisit ditolak untuk lebih dari satu sumber daya Amazon S3
- Semua sumber daya — Kebijakan ini ditetapkan untuk semua sumber daya dalam layanan. Dalam contoh ini, kebijakan ini memungkinkan tindakan terdaftar untuk dilakukan pada semua sumber daya tagihan.
- Teks Sumber Daya – Kebijakan ini mencakup suatu sumber daya yang terdapat dalam layanan. Dalam contoh ini, tindakan yang tercantum hanya diperbolehkan pada `DeploymentGroupName` CodeDeploy sumber daya. Bergantung pada informasi yang disediakan layanan kepada IAM, Anda mungkin melihat ARN atau Anda mungkin melihat jenis sumber daya yang ditentukan.

> **Note**
>
> Kolom ini dapat menyertakan sumber daya dari layanan yang berbeda. Jika pernyataan kebijakan yang mencakup sumber daya tidak mencakup tindakan dan sumber daya dari layanan yang sama, maka kebijakan Anda mencakup sumber daya yang tidak sesuai. IAM tidak memperingatkan Anda tentang sumber daya yang tidak sesuai ketika Anda membuat kebijakan, atau ketika Anda melihat kebijakan dalam ringkasan kebijakan. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Untuk lebih memahami kebijakan Anda, selalu uji kebijakan tersebut dengan simulator kebijakan.

E. Minta kondisional – Kolom ini menunjukkan apakah layanan atau tindakan yang terkait dengan sumber daya ditujukan kepada kondisi-kondisi.

- Tidak ada – Kebijakan ini tidak mengikutkan kondisi untuk layanan. Dalam contoh ini tidak ada kondisi yang diterapkan pada tindakan yang ditolak dalam layanan Amazon S3
- Kondisi teks – Kebijakan ini mencakup suatu kondisi untuk layanan. Dalam contoh ini, tindakan Penagihan yang terdaftar hanya diperbolehkan jika alamat IP sumber cocok`203.0.113.0/24`.
- Banyak – Kebijakan ini mencakup lebih dari satu kondisi untuk layanan tersebut. Untuk melihat masing-masing dari beberapa kondisi kebijakan, pilih JSON untuk melihat dokumen kebijakan.

F. Tampilkan layanan yang tersisa — Alihkan tombol ini untuk memperluas tabel untuk menyertakan layanan yang tidak ditentukan oleh kebijakan. Layanan ini ditolak secara implisit (atau ditolak secara default) dalam kebijakan ini. Namun, pernyataan dalam kebijakan lain mungkin masih mengizinkan atau secara eksplisit menolak menggunakan layanan tersebut. Ringkasan kebijakan merangkum izin satu kebijakan. Untuk mempelajari tentang bagaimana AWS layanan memutuskan apakah permintaan yang diberikan harus diizinkan atau ditolak, lihatLogika evaluasi kebijakan.

Ketika kebijakan atau elemen dalam kebijakan tidak memberikan izin, IAM memberikan peringatan dan informasi tambahan dalam ringkasan kebijakan. Tabel ringkasan kebijakan berikut menunjukkan perluasan Tampilkan layanan yang tersisa di halaman detail SummaryAllElementskebijakan dengan kemungkinan peringatan.

Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

Dalam image sebelumnya, Anda dapat melihat semua layanan yang mencakup tindakan, sumber daya, atau kondisi tertentu, tanpa izin:

a. Peringatan sumber daya – Untuk layanan yang tidak memberikan izin untuk semua tindakan atau sumber daya yang disertakan, Anda melihat salah satu upaya berikut di kolom Sumber Daya pada tabel:

- ⚠ Tidak ada sumber daya yang ditentukan. – Ini berarti bahwa layanan telah menentukan tindakan tetapi tidak ada sumber daya didukung yang tercakup dalam kebijakan.

- Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. – Ini berarti bahwa layanan telah menentukan tindakan, tetapi bahwa beberapa tindakan tersebut tidak memiliki sumber daya yang Support.
- Satu atau lebih sumber daya tidak memiliki tindakan yang berlaku. – Ini berarti bahwa layanan telah mendefinisikan sumber daya, tetapi beberapa sumber daya tersebut tidak memiliki tindakan yang mendukung.

Jika layanan mencakup kedua tindakan yang tidak memiliki sumber daya dan sumber daya yang berlaku yang memiliki sumber daya yang berlaku, maka hanya Satu atau lebih sumber daya yang tidak memiliki tindakan yang berlaku. peringatan ditampilkan. Ini karena ketika Anda melihat ringkasan layanan untuk layanan tersebut, sumber daya yang tidak berlaku untuk tindakan apa pun tidak ditampilkan. Untuk tindakan `ListAllMyBuckets`, kebijakan ini mencakup peringatan terakhir karena tindakan tersebut tidak mendukung izin tingkat sumber daya, dan tidak mendukung tombol syarat `s3:x-amz-acl`. Jika Anda memperbaiki masalah sumber daya atau masalah syarat, masalah yang tersisa muncul di peringatan mendetail.

b. Minta peringatan kondisional – Untuk layanan yang tidak memberikan izin untuk semua kondisi yang disertakan, Anda melihat salah satu peringatan berikut dalam kolom Minta kondisional pada tabel:

- Satu atau lebih tindakan tidak memiliki kondisi yang berlaku. – Ini berarti bahwa layanan telah menentukan tindakan, tetapi beberapa tindakan tersebut tidak memiliki kondisi yang mendukung
- Satu atau lebih kondisi tidak memiliki tindakan yang berlaku. – Ini berarti bahwa layanan telah memiliki kondisi yang ditetapkan, tetapi sebagian dari kondisi tersebut tidak memiliki tindakan yang mendukung.

c. Beberapa |

Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. – `Deny` pernyataan untuk Amazon S3 mencakup lebih dari satu sumber daya. Ia juga mencakup lebih dari satu tindakan, dan beberapa tindakan mendukung sumber daya dan beberapa lainnya tidak. Untuk melihat kebijakan ini, lihat the section called “SummaryAllElements Dokumen kebijakan JSON”. Dalam hal

ini, kebijakan ini mencakup semua tindakan Amazon S3 dan hanya tindakan yang dapat dilakukan pada bucket atau objek bucket yang ditolak.

d. ⚠ Tidak ada sumber daya yang ditentukan — Layanan memiliki tindakan yang ditentukan, tetapi tidak ada sumber daya yang didukung yang disertakan dalam kebijakan, dan oleh karena itu layanan tidak memberikan izin. Dalam hal ini, kebijakan mencakup CodeCommit tindakan tetapi tidak ada CodeCommit sumber daya.

e. DeploymentGroupName | string seperti | Semua, wilayah | string seperti | us-west-2

⚠ | Satu atau lebih tindakan tidak memiliki sumber daya yang berlaku. — Layanan memiliki tindakan yang ditentukan, dan setidaknya satu tindakan lagi yang tidak memiliki sumber daya pendukung.

f. Tidak ada |

⚠ Satu atau lebih kondisi tidak memiliki tindakan yang berlaku. — Layanan memiliki setidaknya satu kunci kondisi yang tidak memiliki tindakan pendukung.

SummaryAllElements Dokumen kebijakan JSON

SummaryAllElements kebijakan ini tidak dimaksudkan untuk Anda gunakan dalam menentukan izin dalam akun Anda. Sebaliknya, ia disertakan untuk menunjukkan kesalahan dan peringatan yang mungkin Anda temukan saat melihat ringkasan kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-
project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
```

```
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Lihat ringkasan kebijakan

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke pengguna atau peran IAM. Untuk kebijakan terkelola, Anda dapat melihat ringkasan kebijakan di halaman Kebijakan. Jika kebijakan Anda tidak menyertakan ringkasan kebijakan, lihat Ringkasan kebijakan hilang untuk mempelajari alasannya.

Melihat ringkasan kebijakan dari halaman Kebijakan

Anda dapat melihat ringkasan kebijakan untuk kebijakan yang dikelola pada halaman Kebijakan.

Untuk melihat ringkasan kebijakan dari halaman Kebijakan

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.

3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.

Melihat ringkasan kebijakan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke pengguna IAM.

Untuk melihat ringkasan kebijakan yang terlampir pada pengguna

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.
5. Dalam tabel kebijakan untuk pengguna, perluas baris kebijakan yang ingin Anda lihat.

Melihat ringkasan kebijakan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke peran.

Untuk melihat ringkasan kebijakan yang terlampir pada peran

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk peran, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada peran.
5. Dalam tabel kebijakan untuk peran tersebut, perluas baris kebijakan yang ingin Anda lihat.

Menyunting kebijakan untuk memperbaiki peringatan

Saat melihat ringkasan kebijakan, Anda mungkin menemukan kesalahan ketik atau pemberitahuan bahwa kebijakan tersebut tidak memberikan izin yang Anda harapkan. Anda tidak dapat langsung menyunting ringkasan kebijakan. Namun demikian, Anda dapat menyunting kebijakan terkelola

pelanggan dengan menggunakan editor kebijakan visual, yang menangkap banyak kesalahan dan peringatan yang dilaporkan oleh ringkasan kebijakan. Kemudian, Anda dapat melihat perubahan dalam ringkasan kebijakan untuk mengonfirmasi bahwa Anda sudah menyelesaikan semua masalah. Untuk mempelajari cara sunting kebijakan selaras, lihat the section called “Edit kebijakan IAM”. Anda tidak dapat mengedit kebijakan AWS terkelola.

Anda dapat mengedit kebijakan untuk ringkasan kebijakan Anda menggunakan opsi Visual.

Untuk mengedit kebijakan ringkasan kebijakan Anda menggunakan opsi Visual

1. Buka ringkasan kebijakan seperti yang dijelaskan dalam prosedur sebelumnya.
2. Pilih Edit.

   Jika Anda berada di halaman Pengguna dan memilih untuk menyunting kebijakan terkelola pelanggan yang terlampir pada pengguna tersebut, Anda akan diarahkan kembali ke halaman Kebijakan. Anda dapat menyunting kebijakan yang terkelola pelanggan hanya di halaman Kebijakan.
3. Pilih opsi Visual untuk melihat representasi visual kebijakan yang dapat diedit. IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual dan menjadikan Anda lebih mudah untuk menemukan dan membenahi adanya masalah. Peringatan dan pesan adanya kesalahan di halaman dapat memandu Anda untuk membenahi adanya masalah dalam kebijakan Anda. Untuk informasi selengkapnya tentang kebijakan berbasis sumber daya, lihat Restrukturisasi kebijakan.
4. Edit kebijakan Anda dan pilih Berikutnya untuk melihat perubahan yang tercermin dalam ringkasan kebijakan. Jika Anda masih melihat masalah, pilih Sebelumnya untuk kembali ke layar penyuntingan.
5. Pilih Simpan perubahan untuk menyimpan perubahan Anda.

Anda dapat mengedit kebijakan untuk ringkasan kebijakan menggunakan opsi JSON.

Untuk mengedit kebijakan ringkasan kebijakan Anda menggunakan opsi JSON

1. Buka ringkasan kebijakan seperti yang dijelaskan dalam prosedur sebelumnya.
2. Anda dapat menggunakan tombol Ringkasan dan JSON untuk membandingkan ringkasan kebijakan dengan dokumen kebijakan JSON. Anda dapat menggunakan informasi ini untuk menentukan baris mana dalam dokumen kebijakan yang ingin Anda ubah.
3. Pilih Edit dan kemudian pilih opsi JSON untuk mengedit dokumen kebijakan JSON.

> **Note**
>
> Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di opsi Editor Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

Jika Anda berada di halaman Pengguna dan memilih untuk menyunting kebijakan terkelola pelanggan yang terlampir pada pengguna tersebut, Anda akan diarahkan kembali ke halaman Kebijakan. Anda dapat menyunting kebijakan yang terkelola pelanggan hanya di halaman Kebijakan.

4. Edit kebijakan Anda. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya. Jika Anda masih melihat masalah, pilih Sebelumnya untuk kembali ke layar penyuntingan.
5. Pilih Simpan perubahan untuk menyimpan perubahan Anda.

Tingkat akses dalam ringkasan kebijakan

AWS ringkasan tingkat akses

Ringkasan kebijakan mencakup ringkasan tingkat akses yang menjelaskan izin tindakan yang ditentukan untuk setiap layanan yang disebutkan dalam kebijakan. Untuk ARN ringkasan kebijakan, lihat Ringkasan kebijakan. Ringkasan tingkat akses menunjukkan apakah tindakan di setiap tingkat akses (`List`,,`Read`, `TaggingWrite`, dan`Permissions management`) memiliki `Full` atau `Limited` izin yang ditentukan dalam kebijakan. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Contoh berikut menjelaskan akses yang diberikan oleh kebijakan untuk layanan tertentu. Untuk contoh-contoh dokumen kebijakan JSON lengkap dan rangkuman terkait, lihat Contoh ringkasan kebijakan.

Layanan	Tingkat akses	Kebijakan memungkinkan hal-hal berikut
IAM	Akses penuh	Akses ke semua tindakan dalam layanan IAM.

Layanan	Tingkat akses	Kebijakan memungkinkan hal-hal berikut
CloudWatch	Penuh:Daftar	Akses ke semua CloudWatch tindakan di tingkat `List` akses, tetapi tidak ada akses ke tindakan dengan klasifikasi `ReadWrite`,, atau tingkat `Permissions management` akses.
Data Pipeline	Terbatas: Daftar, Baca	Akses ke setidaknya satu tetapi tidak semua AWS Data Pipeline tindakan di tingkat `Read` akses `List` dan, tetapi bukan `Permissions management` tindakan `Write` atau.
EC2	Penuh:Daftar, Baca Terbatas: Tulis	Akses ke semua Amazon EC2 `List` dan tindakan `Read` serta akses ke setidaknya satu tetapi tidak semua tindakan `Write` Amazon EC2, tetapi tidak ada akses ke tindakan dengan `Permissions management` klasifikasi tingkat akses.
S3	Terbatas: Manajemen Baca, Tulis, Izin	Akses ke setidaknya satu tetapi tidak semua Amazon S3 `Read`, `Write` dan tindakan `Permissions management` .
codedploy	(kosong)	Akses tak dikenal, karena IAM tidak mengenali layanan ini.
API Gateway	Tidak ada	Tidak ada akses yang didefinisikan dalam kebijakan.
CodeBuild	⚠ Tidak ada tindakan yang ditentukan.	Tidak ada akses karena tidak ada tindakan yang ditentukan untuk layanan tersebut. Untuk mempelajari cara memahami dan mengatasi masalah ini, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.

Dalam ringkasan kebijakan, Ak ses penuh menunjukkan bahwa kebijakan menyediakan akses ke semua tindakan dalam layanan. Kebijakan yang memberikan akses ke beberapa tetapi tidak semua

tindakan dalam layanan akan dikelompokkan lebih lanjut sesuai dengan klasifikasi tingkat akses. Hal ini ditunjukkan oleh salah satu pengelompokan tingkat akses berikut:

- Penuh: Kebijakan tersebut memberikan akses ke semua tindakan dalam klasifikasi tingkat akses yang ditentukan.
- Terbatas: Kebijakan tersebut memberikan akses ke satu atau lebih tetapi tidak semua tindakan dalam klasifikasi tingkat akses yang ditentukan.
- TIdak Ada: Kebijakan ini tidak memiliki akses.
- (kosong): IAM tidak mengenali layanan ini. Jika nama layanan mencakup salah ketik, maka kebijakan tersebut tidak memberikan akses ke layanan. Jika nama layanan sudah benar, maka layanan mungkin tidak mendukung rangkuman kebijakan atau mungkin sedang ditampilkan di pratinjau. Dalam hal ini, kebijakan mungkin memberikan akses, tetapi akses tersebut tidak dapat ditampilkan dalam ringkasan kebijakan. Untuk meminta dukungan ringkasan kebijakan untuk layanan yang tersedia secara umum (GA), lihat Layanan tidak mendukung ringkasan kebijakan IAM.

Ringkasan tingkat akses yang mencakup akses terbatas (sebagian) ke tindakan dikelompokkan menggunakan klasifikasi tingkat AWS aksesList,,, Read TaggingWrite, atau. Permissions management

AWS tingkat akses

AWS mendefinisikan klasifikasi tingkat akses berikut untuk tindakan dalam layanan:

- Daftar: Izin untuk mencantumkan sumber daya di dalam layanan untuk menentukan apakah ada objek. Tindakan dengan tingkat akses ini dapat mencantumkan objek tetapi tidak dapat melihat isi sumber daya. Misalnya, tindakan Amazon S3 `ListBucket` memiliki tingkat akses Daftar.
- Baca: Izin untuk membaca, namun tidak mengedit konten dan atribut sumber daya dalam layanan. Misalnya, tindakan Amazon S3 `GetObject` and `GetBucketLocation` memiliki tingkat akses Baca.
- Penandaan: Izin untuk melakukan tindakan yang hanya mengubah status tanda sumber daya. Misalnya, tindakan IAM `TagRole` dan `UntagRole` memiliki tingkat akses Penandaan karena hanya mengizinkan tindakan untuk menandai atau menghapus peran. Namun, tindakan `CreateRole` memungkinkan menandai sumber daya peran saat Anda membuat peran tersebut. Karena tindakan tersebut tidak hanya menambahkan tanda, tindakan memiliki tingkat akses `Write`.

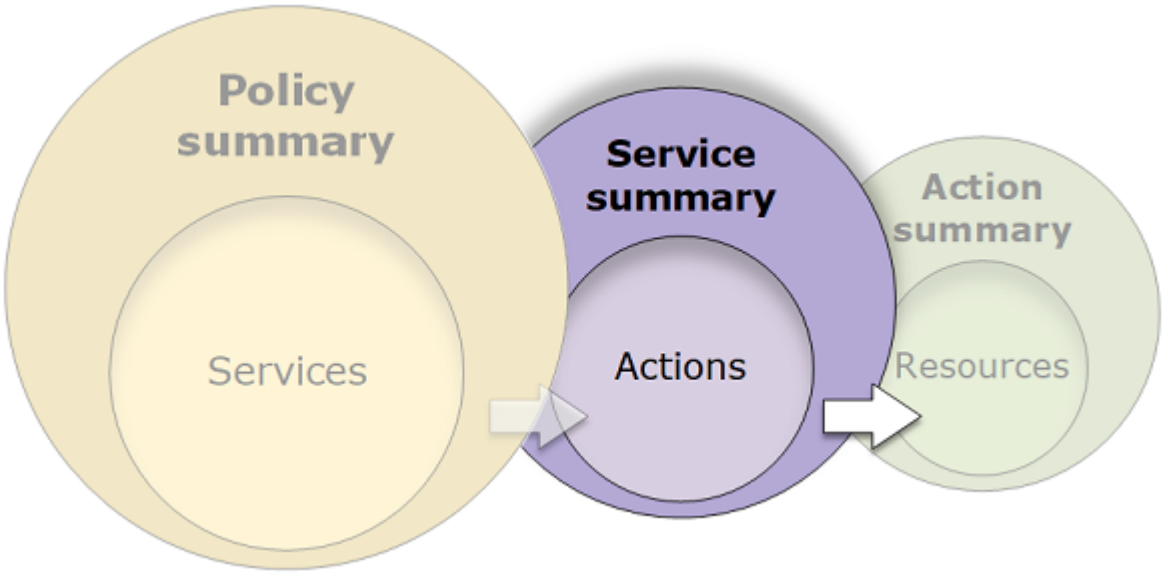
- Tulis: Izin untuk membuat, menghapus, atau memodifikasi sumber daya dalam layanan. Misalnya, tindakan Amazon S3`CreateBucket`, `DeleteBucket` dan `PutObject` memiliki tingkat akses T ulis. `Write`tindakan mungkin juga memungkinkan memodifikasi tag sumber daya. Namun, tindakan yang hanya memungkinkan perubahan pada tanda memiliki tingkat akses `Tagging`.
- Manajemen izin: Manajemen izin mengacu pada tindakan yang mengontrol akses di dalam Layanan AWS, termasuk izin identitas IAM dan non-IAM, tetapi tidak termasuk kontrol akses tingkat jaringan seperti grup keamanan. Misalnya, sebagian besar IAM dan AWS Organizations tindakan, serta tindakan Amazon S3 `PutBucketPolicy` dan `DeleteBucketPolicy` memiliki tingkat akses manajemen Izin.

> **Kiat**
>
> Untuk meningkatkan keamanan Anda Akun AWS, batasi atau pantau secara teratur kebijakan yang menyertakan klasifikasi tingkat akses manajemen izin.

Untuk melihat klasifikasi tingkat akses untuk semua tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Ringkasan layanan (daftar tindakan)

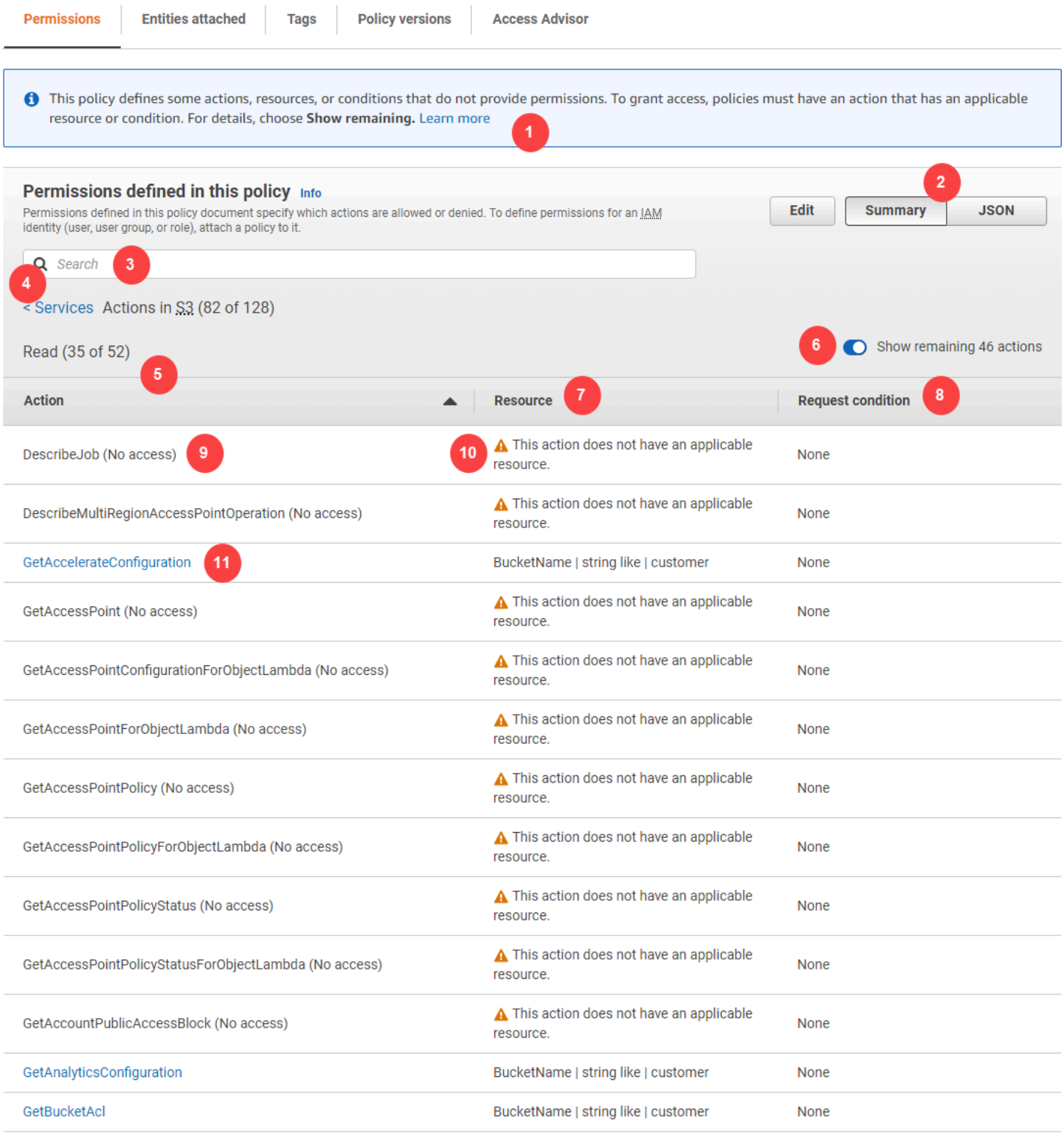
Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan layanan menyertakan daftar tindakan dan ringkasan izin yang ditentukan oleh kebijakan untuk layanan yang dipilih.



Anda dapat melihat ringkasan layanan untuk setiap layanan yang tercantum dalam ringkasan kebijakan yang memberikan izin. Tabel dikelompokkan ke dalam Tindakan yang tidak dikategorikan, Jenis sumber daya yang tidak dikategorikan, dan bagian tingkat akses. Jika kebijakan mencakup tindakan yang tidak dikenali oleh IAM, maka tindakan tersebut disertakan dalam bagian Tindakan yang tidak dikategorikan di tabel. Jika IAM mengenali tindakan, maka itu termasuk dalam salah satu bagian tingkat akses (Daftar, Baca, Tulis dan Manajemen izin) pada tabel. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Memahami elemen ringkasan layanan

Contoh di bawah ini adalah ringkasan layanan untuk tindakan Amazon S3 yang diizinkan dari ringkasan kebijakan. Tindakan untuk layanan ini dikelompokkan berdasarkan tingkat akses. Misalnya, 35 tindakan Baca ditentukan dari total 52 tindakan Baca yang tersedia untuk layanan.



Halaman ringkasan layanan untuk kebijakan yang dikelola mencakup informasi berikut ini:

1. Jika kebijakan tidak memberikan izin untuk semua tindakan, sumber daya, dan kondisi yang ditentukan untuk layanan dalam kebijakan, banner peringatan akan muncul di bagian atas

halaman. Ringkasan layanan kemudian mencakup rincian tentang masalah. Untuk mempelajari bagaimana rangkuman kebijakan membantu Anda memahami dan mengatasi masalah izin yang diberikan oleh kebijakan Anda, lihat the section called “Kebijakan saya tidak memberikan izin yang diharapkan”.

2. Pilih JSON untuk melihat detail tambahan tentang kebijakan. Anda dapat melakukan ini untuk melihat semua kondisi yang diterapkan ke tindakan. (Jika Anda melihat ringkasan layanan untuk kebijakan inline yang dilampirkan langsung ke pengguna, Anda harus menutup kotak dialog ringkasan layanan dan kembali ke ringkasan kebijakan untuk mengakses dokumen kebijakan JSON.)
3. Untuk melihat ringkasan tindakan tertentu, ketik kata kunci ke dalam kotak Pencarian untuk mengurangi daftar tindakan yang tersedia.
4. Di sebelah panah kembali Layanan muncul nama layanan (dalam hal ini S3). Ringkasan layanan untuk layanan ini mencakup daftar tindakan yang diizinkan atau ditolak yang ditentukan dalam kebijakan. Jika layanan muncul di bawah (Penolakan eksplisit) pada tab Izin, maka tindakan yang tercantum dalam tabel ringkasan layanan ditolak secara eksplisit. Jika layanan muncul di bawah Izinkan pada tab Izin, maka tindakan yang tercantum dalam tabel ringkasan layanan diizinkan.
5. T indakan — Kolom ini mencantumkan tindakan yang ditentukan dalam kebijakan dan menyediakan sumber daya dan kondisi untuk setiap tindakan. Jika kebijakan memberikan atau menolak izin untuk tindakan, maka nama tindakan akan menautkan ke tabel ringkasan tindakan. Tabel mengelompokkan tindakan ini menjadi setidaknya satu atau hingga lima bagian, tergantung pada tingkat akses yang diizinkan atau ditolak oleh kebijakan. Bagian-bagiannya adalah Daftar, Baca, T ulis, Manajemen Izin, dan Pen andaan. Hitungan menunjukkan jumlah tindakan yang dikenali yang memberikan izin dalam setiap tingkat akses. Totalnya adalah jumlah tindakan yang diketahui untuk layanan tersebut. Dalam contoh ini, 35 tindakan memberikan izin dari total 52 tindakan Baca Amazon S3 yang diketahui. Untuk melihat klasifikasi tingkat akses yang ditetapkan untuk setiap tindakan dalam layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.
6. Tampilkan tindakan yang tersisa — Alihkan tombol ini untuk memperluas atau menyembunyikan tabel untuk menyertakan tindakan yang diketahui tetapi tidak memberikan izin untuk layanan ini. Mengalihkan tombol juga menampilkan peringatan untuk elemen apa pun yang tidak memberikan izin.
7. Sumber Daya – Kolom ini menunjukkan sumber daya yang ditentukan oleh kebijakan untuk layanan. IAM tidak memeriksa jika sumber daya berlaku untuk setiap tindakan. Dalam contoh ini, tindakan dalam layanan Amazon S3 hanya diperbolehkan pada sumber daya bucket

`developer_bucket` Amazon S3. Bergantung pada informasi yang disediakan layanan untuk IAM, Anda mungkin melihat ARN seperti `arn:aws:s3:::developer_bucket/*`, atau Anda mungkin melihat jenis sumber daya yang ditentukan, seperti `BucketName = developer_bucket`.

> **Note**
>
> Kolom ini dapat menyertakan sumber daya dari layanan yang berbeda. Jika pernyataan kebijakan yang mencakup sumber daya tidak mencakup tindakan dan sumber daya dari layanan yang sama, maka kebijakan Anda mencakup sumber daya yang tidak sesuai. IAM tidak memperingatkan Anda tentang sumber daya yang tidak cocok saat Anda membuat kebijakan, atau saat Anda melihat kebijakan di ringkasan layanan. IAM juga tidak menunjukkan apakah tindakan tersebut berlaku untuk sumber daya, hanya jika layanan sesuai. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Untuk lebih memahami kebijakan Anda, selalu uji kebijakan tersebut dengan simulator kebijakan.

8. Minta ketentuan – Kolom ini memberi tahu apakah tindakan yang terkait dengan sumber daya tunduk pada kondisi. Untuk mempelajari lebih lanjut tentang kondisi tersebut, pilih JSON untuk meninjau dokumen kebijakan JSON.
9. (Tanpa akses) – Kebijakan ini mencakup tindakan yang tidak menyediakan izin.
10. Peringatan sumber daya – Untuk tindakan dengan sumber daya yang tidak memberikan izin penuh, Anda akan melihat salah satu dari peringatan berikut:
    - Tindakan ini tidak mendukung izin tingkat sumber daya. Ini memerlukan wildcard (*) untuk sumber daya. – Artinya, kebijakan tersebut menyertakan izin tingkat sumber daya, tetapi harus menyertakan `"Resource": ["*"]` untuk memberikan izin untuk tindakan ini.
    - Tindakan ini tidak memiliki sumber daya yang berlaku. - Artinya tindakan tersebut termasuk dalam kebijakan tanpa sumber daya yang didukung.
    - Tindakan ini tidak memiliki sumber daya dan kondisi yang berlaku. - Artinya tindakan tersebut termasuk dalam kebijakan tanpa sumber daya yang didukung dan tanpa kondisi yang didukung. Dalam hal ini, ada juga ketentuan yang termasuk dalam kebijakan untuk layanan ini, tetapi tidak ada ketentuan yang berlaku untuk tindakan ini.
11. Tindakan yang memberikan izin meliputi tautan ke ringkasan tindakan.

Lihat ringkasan layanan

Anda dapat melihat ringkasan layanan untuk setiap layanan yang tercantum dalam ringkasan kebijakan yang memberikan izin.

Melihat ringkasan layanan dari halaman Kebijakan

Anda dapat melihat ringkasan layanan untuk kebijakan terkelola di halaman Kebijakan.

Untuk melihat ringkasan layanan untuk kebijakan yang dikelola

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.
5. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

Melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan layanan untuk setiap kebijakan yang dilampirkan ke pengguna IAM.

Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.
5. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Pengguna dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Pilih Ringkasan. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

> **Note**
>
> Jika kebijakan yang Anda pilih adalah kebijakan inline yang dilampirkan langsung ke pengguna, maka tabel ringkasan layanan muncul. Jika kebijakan tersebut adalah kebijakan inline yang dilampirkan dari suatu grup, maka Anda dibawa ke dokumen kebijakan JSON untuk grup tersebut. Jika kebijakan tersebut adalah kebijakan yang dikelola, maka Anda dibawa ke ringkasan layanan untuk kebijakan tersebut di halaman Kebijakan.

Melihat ringkasan layanan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan kebijakan untuk setiap kebijakan yang dilampirkan ke peran.

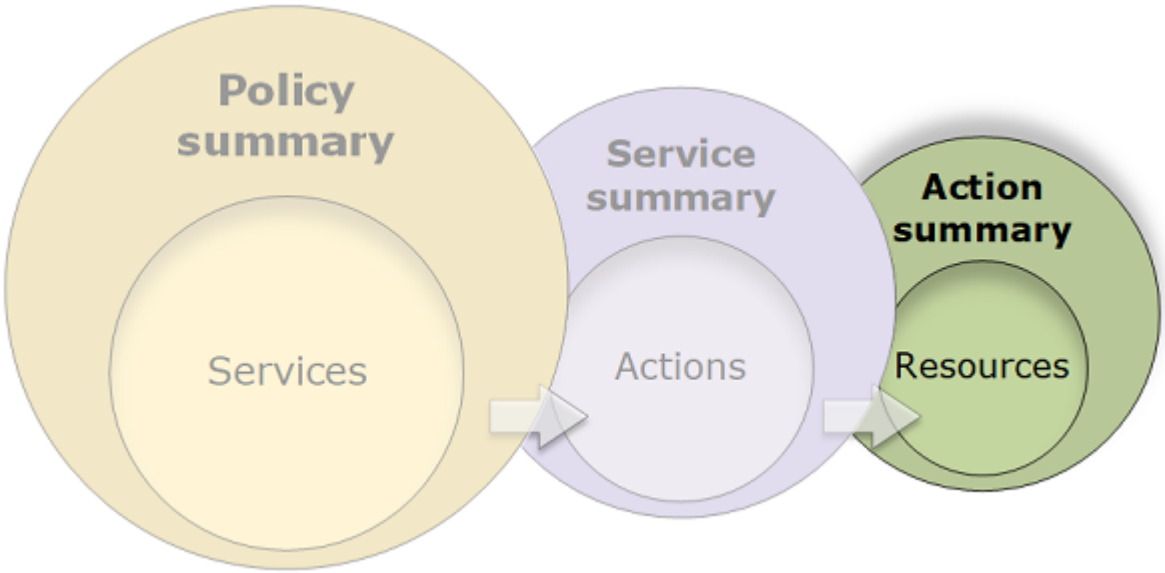
Untuk melihat ringkasan layanan untuk kebijakan yang terlampir pada peran

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Peran dari panel navigasi.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk peran, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada peran.
5. Dalam tabel kebijakan untuk peran tersebut, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Peran dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

Ringkasan tindakan (daftar sumber daya)

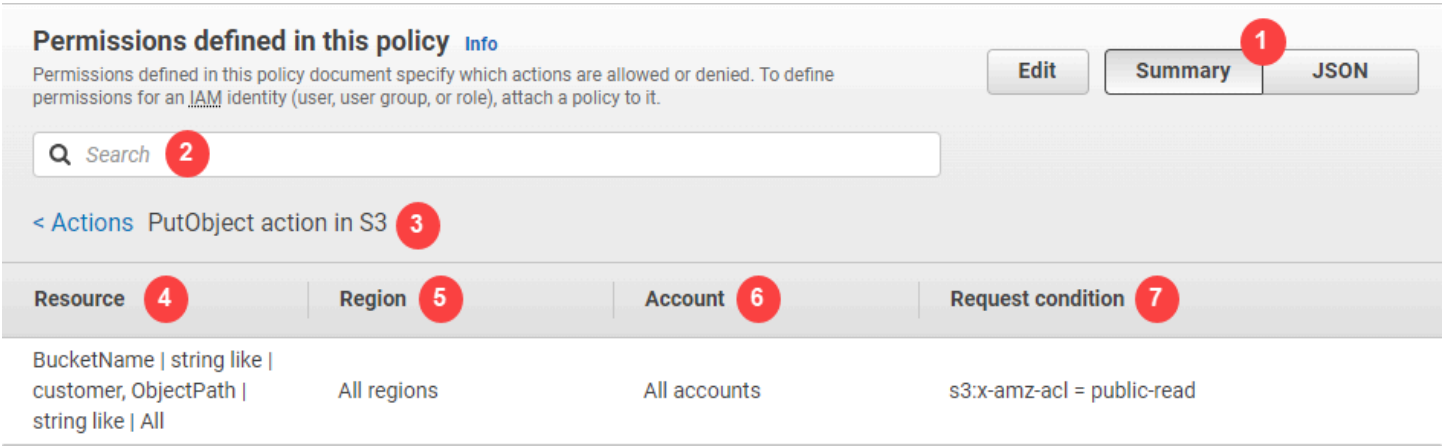
Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan. Tabel ringkasan tindakan mencakup daftar sumber daya dan ketentuan terkait yang berlaku untuk tindakan yang dipilih.



Untuk melihat ringkasan tindakan untuk setiap tindakan yang memberikan izin, pilih tautan dalam ringkasan layanan. Tabel ringkasan tindakan mencakup rincian tentang sumber daya, termasuk Wilayah dan Akunnya. Anda juga dapat melihat ketentuan yang berlaku untuk setiap sumber daya. Hal ini menunjukkan Anda ketentuan yang berlaku untuk beberapa sumber daya tetapi tidak yang lainnya.

Memahami elemen ringkasan tindakan

Contoh di bawah ini adalah ringkasan tindakan untuk tindakan `PutObject` (Tulis) dari ringkasan layanan Amazon S3 (lihat Ringkasan layanan (daftar tindakan)). Untuk tindakan ini, kebijakan ini mendefinisikan beberapa ketentuan pada satu sumber daya.



Halaman ringkasan tindakan mencakup informasi berikut:

1. Pilih JSON untuk melihat detail tambahan tentang kebijakan, seperti melihat beberapa kondisi yang diterapkan pada tindakan. (Jika Anda melihat ringkasan tindakan untuk kebijakan inline yang dilampirkan langsung ke pengguna, langkah-langkahnya berbeda. Untuk mengakses dokumen kebijakan JSON dalam kasus tersebut, Anda harus menutup kotak dialog ringkasan tindakan dan kembali ke ringkasan kebijakan.)
2. Untuk melihat ringkasan sumber daya tertentu, ketik kata kunci ke dalam kotak Pencarian untuk mengurangi daftar sumber daya yang tersedia.
3. Di sebelah Tindakan panah kembali muncul nama layanan dan tindakan dalam format `action name action in service` (dalam hal ini PutObject tindakan di S3). Ringkasan tindakan untuk layanan ini mencakup daftar sumber daya yang ditetapkan dalam kebijakan.
4. Sumber Daya – Kolom ini mencantumkan sumber daya yang ditentukan kebijakan untuk layanan yang dipilih. Dalam contoh ini, tindakan PutObject diperbolehkan di semua jalur objek, tetapi hanya pada sumber daya bucket Amazon S3 `developer_bucket`. Bergantung pada informasi yang disediakan layanan untuk IAM, Anda mungkin melihat ARN seperti `arn:aws:s3:::developer_bucket/*`, atau Anda mungkin melihat jenis sumber daya yang ditentukan, seperti `BucketName = developer_bucket, ObjectPath = All`.
5. Wilayah – Kolom ini menunjukkan Wilayah tempat sumber daya ditetapkan. Sumber daya dapat ditentukan untuk semua Wilayah, atau satu Wilayah. Sumber daya tidak bisa ada di lebih dari satu Wilayah tertentu.
   - Semua wilayah — Tindakan yang terkait dengan sumber daya berlaku untuk semua Wilayah. Dalam contoh ini, tindakan tersebut menjadi milik layanan global, Amazon S3. Tindakan yang termasuk dalam layanan global berlaku untuk semua Wilayah.
   - Teks area – Tindakan terkait dengan sumber daya berlaku untuk satu Wilayah. Misalnya, kebijakan dapat menentukan WIlayah `us-east-2` untuk sumber daya.
6. Akun – Kolom ini menunjukkan apakah layanan atau tindakan yang terkait dengan sumber daya berlaku untuk akun tertentu. Sumber daya dapat muncul di semua akun atau satu akun. Sumber daya tidak bisa ada di lebih dari satu akun tertentu.
   - Semua akun – Tindakan yang terkait dengan sumber daya berlaku untuk semua akun. Dalam contoh ini, tindakan tersebut menjadi milik layanan global, Amazon S3. Tindakan yang termasuk dalam layanan global berlaku untuk semua akun.
   - Akun ini — Tindakan yang terkait dengan sumber daya hanya berlaku di akun saat ini..
   - Nomor akun – Tindakan yang terkait dengan sumber daya berlaku untuk satu akun (akun yang saat ini Anda tidak masuk). Misalnya, jika suatu kebijakan menetapkan akun `123456789012` untuk sumber daya, maka nomor akun muncul di ringkasan kebijakan.

7. Minta ketentuan – Kolom ini menunjukkan apakah tindakan yang terkait dengan sumber daya tunduk pada kondisi. Contoh ini mencakup ketentuan `s3:x-amz-acl = public-read`. Untuk mempelajari lebih lanjut tentang kondisi tersebut, pilih JSON untuk meninjau dokumen kebijakan JSON.

Lihat ringkasan tindakan

Anda dapat melihat ringkasan tindakan untuk setiap tindakan yang tercantum dalam ringkasan kebijakan yang memberikan izin.

Melihat ringkasan tindakan dari halaman Kebijakan

Anda dapat melihat ringkasan tindakan untuk kebijakan terkelola.

Untuk melihat ringkasan tindakan untuk kebijakan yang dikelola

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
4. Pada halaman Detail kebijakan untuk kebijakan, lihat tab Izin untuk melihat ringkasan kebijakan.
5. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.
6. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Melihat ringkasan tindakan untuk kebijakan yang dilampirkan ke pengguna

Anda dapat melihat ringkasan tindakan untuk kebijakan apa pun yang dilampirkan ke pengguna.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada pengguna

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna.
3. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk pengguna, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada pengguna secara langsung atau dari grup.

5. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Pengguna dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.

   > **Note**
   >
   > Jika kebijakan yang Anda pilih adalah kebijakan inline yang dilampirkan langsung ke pengguna, maka tabel ringkasan layanan muncul. Jika kebijakan tersebut adalah kebijakan inline yang dilampirkan dari suatu grup, maka Anda dibawa ke dokumen kebijakan JSON untuk grup tersebut. Jika kebijakan tersebut adalah kebijakan yang dikelola, maka Anda dibawa ke ringkasan layanan untuk kebijakan tersebut di halaman Kebijakan.
7. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Melihat ringkasan tindakan untuk kebijakan yang dilampirkan pada peran

Anda dapat melihat ringkasan tindakan untuk kebijakan apa pun yang dilampirkan ke peran.

Untuk melihat ringkasan tindakan untuk kebijakan yang terlampir pada peran

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Dalam daftar peran, pilih nama peran yang kebijakannya ingin Anda lihat.
4. Di halaman Ringkasan untuk peran, lihat tab Izin untuk melihat daftar kebijakan yang dilampirkan pada peran.
5. Dalam tabel kebijakan untuk peran tersebut, pilih nama kebijakan yang ingin Anda lihat.

   Jika Anda berada di halaman Peran dan memilih untuk melihat ringkasan layanan untuk kebijakan yang dilampirkan ke pengguna tersebut, Anda akan diarahkan ke halaman Kebijakan. Anda dapat melihat ringkasan layanan hanya di halaman Kebijakan.
6. Di daftar ringkasan kebijakan layanan, pilih nama layanan yang ingin Anda lihat.
7. Di daftar tindakan ringkasan layanan, pilih nama tindakan yang ingin Anda lihat.

Contoh ringkasan kebijakan

Contoh berikut mencakup kebijakan JSON dengan ringkasan kebijakan terkait, ringkasan layanan, dan ringkasan tindakan untuk membantu Anda memahami izin yang diberikan melalui kebijakan.

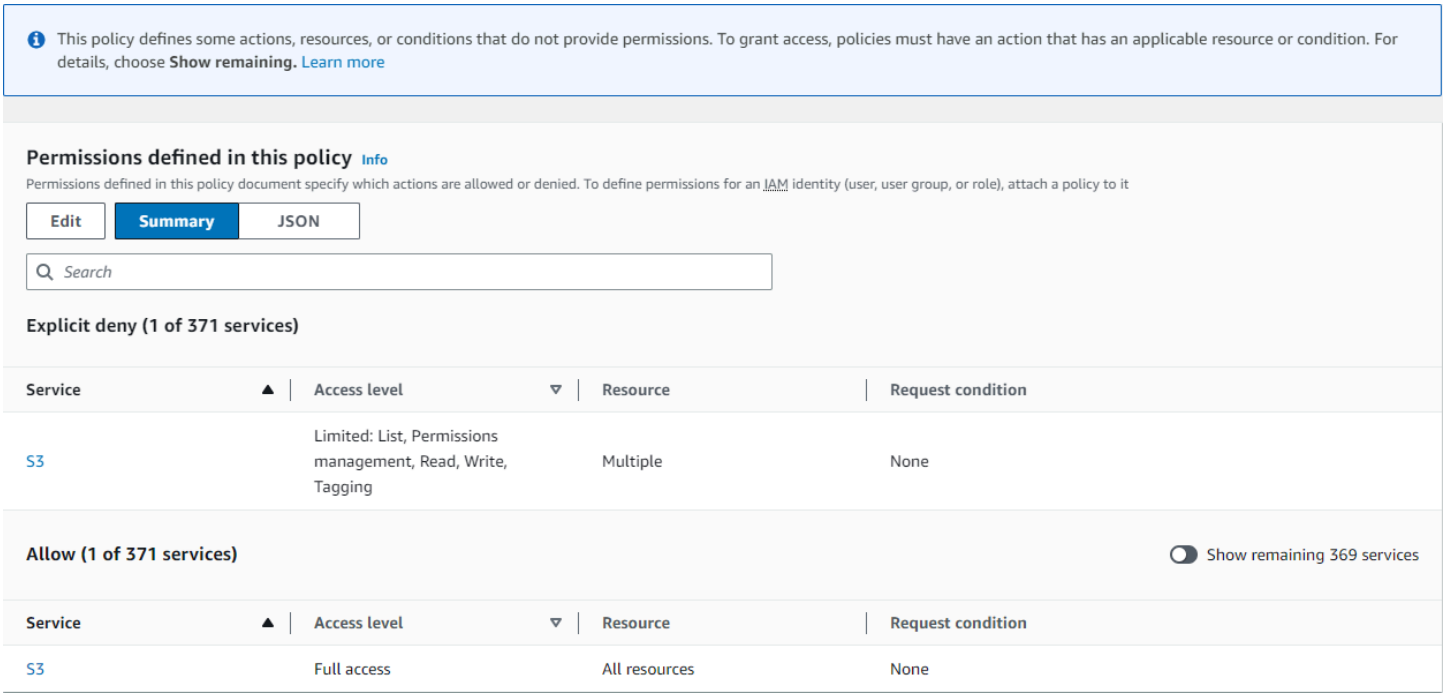
Kebijakan 1: DenyCustomerBucket

Kebijakan ini menunjukkan izin dan penolakan untuk layanan yang sama.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucketRingkasan Kebijakan:



DenyCustomerBucket Ringkasan Layanan S3 (Penolakan eksplisit):

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (Baca) Ringkasan T indakan:

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

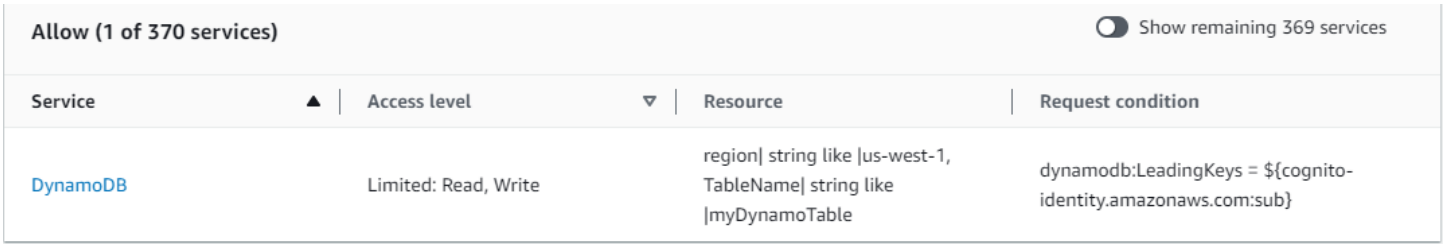
Kebijakan 2: DynamoDbRowCognitoID

Kebijakan ini menyediakan akses tingkat baris ke Amazon DynamoDB berdasarkan ID Amazon Cognito pengguna.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoIDRingkasan Kebijakan:



DynamoDbRowCognitoID Ringkasan Layanan DynamoDB (Izinkan):

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem (Daftar) Ringkasan T indakan:

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Kebijakan 3: MultipleResourceCondition

Kebijakan ini mencakup beberapa sumber daya dan ketentuan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

MultipleResourceConditionRingkasan Kebijakan:

Allow (1 of 370 services) Show remaining 369 services

Service	Access level	Resource	Request condition
S3	Limited: Permissions management, Write	Multiple	Multiple

MultipleResourceCondition Ringkasan Layanan S3 (Izinkan):

< Services Actions in S3 (2 of 130) Show remaining 128 actions

Write (1 of 47)

Action	Resource	Request condition
PutObject	Multiple	Multiple

Permission Management (1 of 15)

Action	Resource	Request condition
PutObjectAcl	Multiple	Multiple

PutObject (Tulis) Ringkasan T indakan:

< Actions PutObject action in S3

Resource	Region	Account	Request condition
Multiple	-	All accounts	Multiple

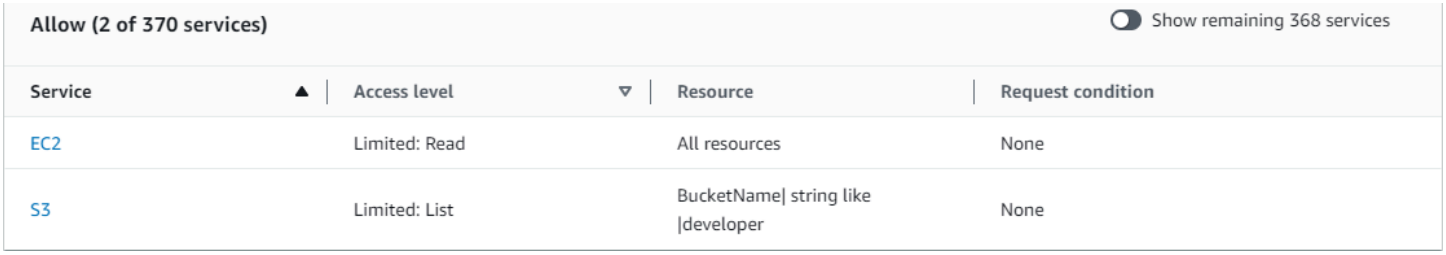
Kebijakan 4: EC2_troubleshoot

Kebijakan berikut memungkinkan pengguna untuk mendapatkan tangkapan layar instans Amazon EC2 yang dapat membantu mengatasi masalah EC2. Kebijakan ini juga mengizinkan melihat informasi tentang item di bucket pengembang Amazon S3.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Ringkasan Kebijakan Troubleshooting:



EC2_Troubleshooting S3 (Izinkan) Ringkasan Layanan:

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

ListBucket (Daftar) Ringkasan T indakan:

< Actions ListBucket action in S3

Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

Kebijakan 5: CodeBuild _ CodeCommit _ CodeDeploy

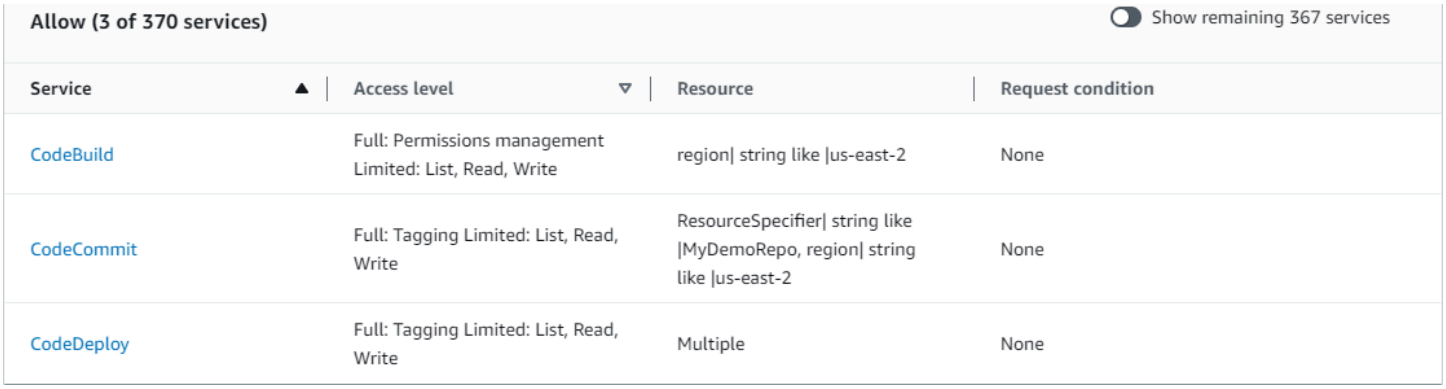
Kebijakan ini menyediakan akses ke CodeDeploy sumber daya tertentu CodeBuild. CodeCommit Karena sumber daya ini khusus untuk setiap layanan, mereka hanya muncul dengan layanan yang cocok. Jika Anda menyertakan sumber daya yang tidak cocok dengan layanan apa pun dalam elemen `Action` tersebut, sumber daya akan muncul di semua ringkasan tindakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-
project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-
east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
```

```
            ]
        }
    ]
}
```

CodeBuild_ CodeCommit _ Ringkasan CodeDeploy Kebijakan:



Allow (3 of 370 services) — Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_ CodeCommit _ CodeDeploy CodeBuild (Izinkan) Ringkasan Layanan:

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_ CodeCommit _ CodeDeploy StartBuild (Tulis) Ringkasan T indakan:

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Izin diperlukan untuk mengakses sumber daya IAM

Sumber Daya adalah objek dalam suatu layanan. Sumber daya IAM meliputi grup, pengguna, peran, dan kebijakan. Jika Anda masuk dengan Pengguna root akun AWS kredensional, Anda tidak memiliki batasan dalam mengelola kredenSIAL IAM atau sumber daya IAM. Namun, pengguna IAM harus secara eksplisit diberikan izin untuk mengelola kredensial atau sumber daya IAM. Anda dapat melakukannya dengan melampirkan kebijakan berbasis identitas kepada pengguna.

> **Note**
>
> Sepanjang AWS dokumentasi, ketika kami merujuk pada kebijakan IAM tanpa menyebutkan kategori tertentu, yang kami maksud adalah kebijakan berbasis identitas dan dikelola pelanggan. Untuk perincian tentang kategori kebijakan, lihat the section called “Kebijakan dan Izin”.

Izin untuk mengurus identitas IAM

Izin yang diperlukan untuk mengelola grup IAM, pengguna, peran, dan kredensial yang biasanya sesuai dengan tindakan API untuk tugas tersebut. Misalnya, untuk membuat pengguna IAM, Anda harus memiliki `iam:CreateUser` izin yang memiliki perintah API yang sesuai: `CreateUser`. Untuk mengizinkan pengguna IAM membuat pengguna IAM lain, Anda dapat melampirkan kebijakan IAM seperti berikut ini ke pengguna tersebut:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Dalam kebijakan, nilai elemen `Resource` bergantung pada tindakan dan sumber daya yang dapat dipengaruhi oleh tindakan tersebut. Dalam contoh sebelumnya, kebijakan ini memungkinkan

pengguna untuk membuat pengguna mana pun (* adalah wildcard yang cocok dengan semua string). Sebaliknya, kebijakan yang mengizinkan pengguna untuk hanya mengubah access key mereka sendiri (tindakan API `CreateAccessKey` dan `UpdateAccessKey`) biasanya memiliki elemen `Resource`. Dalam hal ini ARN mencakup variabel (`${aws:username}`) yang menyelesaikan nama pengguna saat ini, seperti dalam contoh berikut:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Dalam contoh sebelumnya, `${aws:username}` adalah variabel yang menyelesaikan nama pengguna saat ini. Untuk informasi lebih lanjut tentang variabel-variable kebijakan, lihat Elemen kebijakan IAM: Variabel dan tag.

Menggunakan karakter wildcard (*) dalam nama tindakan sering kali mempermudah pemberian izin untuk semua tindakan yang terkait dengan tugas tertentu. Misalnya, untuk memungkinkan pengguna melakukan tindakan IAM, Anda dapat menggunakan `iam:*` untuk tindakannya. Agar pengguna dapat melakukan tindakan terkait hanya untuk mengakses kunci, Anda dapat menggunakan `iam:*AccessKey*` di elemen `Action` pernyataan kebijakan. Hal ini memberi pengguna izin

untuk melakukan tindakan `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys`, dan `UpdateAccessKey`. (Jika tindakan ditambahkan ke IAM di masa mendatang yang memiliki "AccessKey" dalam nama, menggunakan `iam:*AccessKey*` untuk `Action` elemen juga akan memberikan izin pengguna untuk tindakan baru itu.) Contoh berikut menunjukkan kebijakan yang memungkinkan pengguna melakukan semua tindakan yang berkaitan dengan kunci akses mereka sendiri (ganti *`account-id`* dengan Akun AWS ID Anda):

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/${aws:username}"
    }
}
```

Beberapa tugas, seperti menghapus grup, melibatkan beberapa tindakan: Anda terlebih dahulu harus menghapus pengguna dari grup, kemudian memisahkan atau menghapus kebijakan grup, dan kemudian menghapusnya. Jika Anda ingin pengguna dapat menghapus grup, Anda harus yakin untuk memberikan izin kepada pengguna untuk melakukan semua tindakan terkait.

Izin untuk bekerja di Konsol Manajemen AWS

Contoh-contoh sebelumnya menunjukkan kebijakan yang memungkinkan pengguna untuk melakukan tindakan dengan AWS CLI atau SDK AWS.

Saat pengguna bekerja dengan konsol, konsol mengeluarkan permintaan ke IAM untuk mencantumkan grup, pengguna, peran, dan kebijakan, dan untuk mendapatkan kebijakan yang terkait dengan grup, pengguna, atau peran. Konsol juga mengeluarkan permintaan untuk mendapatkan Akun AWS informasi dan informasi tentang kepala sekolah. Prinsipalnya adalah pengguna yang mengajukan permintaan di konsol.

Secara umum, untuk melakukan tindakan, Anda hanya boleh memiliki tindakan yang sesuai yang termasuk dalam kebijakan. Untuk membuat pengguna, Anda perlu izin untuk menghubungi tindakan `CreateUser`. Sering kali, ketika Anda menggunakan konsol untuk melakukan tindakan, Anda harus memiliki izin untuk menampilkan, membuat daftar, mendapatkan, atau melihat sumber daya

di konsol. Ini diperlukan agar Anda dapat menavigasi melalui konsol untuk membuat tindakan yang ditentukan. Misalnya, jika pengguna Jorge ingin menggunakan konsol untuk mengubah kunci aksesnya sendiri, dia beralih ke konsol IAM dan memilih Pengguna. Tindakan ini menyebabkan konsol untuk membuat permintaan `ListUsers`. Jika Jorge tidak memiliki izin untuk tindakan `iam:ListUsers`, konsol ditolak akses ketika mencoba untuk membuat daftar pengguna. Sebagai hasilnya, Jorge tidak bisa mendapatkan nama dan access key-nya sendiri, meskipun dia memiliki izin untuk tindakan `CreateAccessKey` dan `UpdateAccessKey`.

Jika Anda ingin memberikan izin kepada pengguna untuk mengelola grup, pengguna, peran, kebijakan, dan kredenSIAL dengan Konsol Manajemen AWS, Anda harus menyertakan izin untuk tindakan yang dilakukan konsol. Untuk beberapa contoh kebijakan yang dapat Anda gunakan untuk memberikan izin kepada pengguna, lihat Contoh kebijakan untuk mengelola sumber daya IAM.

Berikan izin di seluruh AWS rekening

Anda dapat memberi izin pengguna IAM secara langsung di akses akun Anda sendiri ke sumber daya Anda. Jika pengguna dari akun lain memerlukan akses ke sumber daya Anda, Anda dapat membuat peran IAM, yang merupakan entitas yang menyertakan izin tetapi tidak terkait dengan pengguna tertentu. Pengguna dari akun lain kemudian dapat menggunakan peran dan mengakses sumber daya sesuai dengan izin yang Anda tetapkan untuk peran tersebut. Untuk informasi selengkapnya, lihat Akses untuk pengguna IAM di yang lain Akun AWS yang Anda miliki.

> **Note**
>
> Beberapa layanan mendukung kebijakan berbasis sumber daya sebagaimana dijelaskan dalam Identity-based kebijakan dan kebijakan berbasis sumber daya (seperti Amazon S3, Amazon SNS, dan Amazon SQS). Untuk layanan tersebut, alternatif untuk menggunakan peran adalah dengan memberikan kebijakan ke sumber daya (bucket, topik, atau antrean) yang ingin Anda bagikan. Kebijakan berbasis sumber daya dapat menentukan AWS akun yang memiliki izin untuk mengakses sumber daya.

Izin satu layanan untuk mengakses layanan lainnya

Banyak AWS layanan mengakses AWS layanan lain. Misalnya, beberapa layanan AWS —termasuk Amazon EMR, Elastic Load Balancing, dan Amazon EC2 Auto Scaling—mengelola instans Amazon EC2. AWS Layanan lain menggunakan bucket Amazon S3, topik Amazon SNS, antrian Amazon SQS, dan sebagainya.

Skenario untuk mengelola izin dalam kasus-kasus ini berbeda-beda menurut layanan. Berikut ini beberapa contoh cara izin ditangani untuk layanan yang berbeda:

- Di Amazon EC2 Auto Scaling, pengguna harus memiliki izin untuk menggunakan Auto Scaling, tetapi tidak perlu mendapatkan izin secara eksplisit untuk mengelola instans Amazon EC2.
- Di AWS Data Pipeline, peran IAM menentukan apa yang dapat dilakukan pipeline; pengguna memerlukan izin untuk mengambil peran tersebut. (Untuk detailnya, lihat Mem berikan Izin ke Pipeline dengan IAM di Panduan Peng AWS Data Pipeline embang.)

Untuk detail tentang cara mengonfigurasi izin dengan benar sehingga AWS layanan dapat menyelesaikan tugas yang Anda inginkan, lihat dokumentasi untuk layanan yang Anda panggil. Untuk mempelajari cara membuat peran untuk layanan, lihat Buat peran untuk mendelegasikan izin ke AWS layanan.

Mengonfigurasi layanan dengan peran IAM untuk bekerja atas nama Anda

Ketika Anda ingin mengonfigurasi AWS layanan untuk bekerja atas nama Anda, Anda biasanya menyediakan ARN untuk peran IAM yang menentukan apa yang diizinkan untuk dilakukan layanan. AWS memeriksa untuk memastikan bahwa Anda memiliki izin untuk meneruskan peran ke layanan. Untuk informasi selengkapnya, lihat Memberikan izin pengguna untuk meneruskan peran ke AWS layanan.

Tindakan yang diperlukan

Tindakan adalah hal-hal yang dapat Anda lakukan ke sumber daya, seperti melihat, membuat, mengedit, dan menghapus sumber daya tersebut. Tindakan ditentukan oleh setiap AWS layanan.

Untuk mengizinkan seseorang melakukan suatu tindakan, Anda harus menyertakan tindakan yang diperlukan dalam kebijakan yang berlaku untuk identitas panggilan atau sumber daya yang terpengaruh. Secara umum, untuk memberikan izin yang diperlukan untuk melakukan tindakan, Anda harus menyertakan tindakan tersebut dalam kebijakan Anda. Misalnya, untuk membuat pengguna, Anda perlu menambahkan CreateUser tindakan ke kebijakan Anda.

Dalam beberapa kasus, suatu tindakan mungkin mengharuskan Anda menyertakan tindakan terkait tambahan dalam kebijakan Anda. Misalnya, untuk memberikan izin kepada seseorang untuk membuat direktori di AWS Directory Service menggunakan operasi `ds:CreateDirectory`, Anda harus menyertakan tindakan berikut dalam kebijakan mereka:

- `ds:CreateDirectory`

- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Saat Anda membuat atau mengedit kebijakan menggunakan editor visual, Anda menerima peringatan dan petunjuk untuk membantu Anda memilih semua tindakan yang diperlukan untuk kebijakan Anda.

Untuk informasi selengkapnya tentang izin yang diperlukan untuk membuat direktori di AWS Directory Service, lihat Contoh 2: Izinkan Pengguna Membuat Direktori.

Contoh kebijakan untuk mengelola sumber daya IAM

Berikut adalah contoh kebijakan IAM yang memungkinkan pengguna melakukan tugas yang terkait dengan pengelolaan pengguna IAM, grup, dan kredensial. Ini termasuk kebijakan yang mengizinkan pengguna mengelola kata sandi, kunci akses, dan perangkat multi-factor authentication (MFA) mereka sendiri.

Untuk contoh kebijakan yang memungkinkan pengguna melakukan tugas dengan AWS layanan lain, seperti Amazon S3, Amazon EC2, dan DynamoDB, lihat. Contoh kebijakan berbasis identitas IAM

Topik

- Memungkinkan pengguna untuk membuat daftar grup akun, pengguna, kebijakan, dan lainnya untuk tujuan pelaporan
- Memungkinkan pengguna untuk mengelola keanggotaan grup
- Izinkan pengguna untuk mengelola pengguna IAM
- Izinkan pengguna mengatur kebijakan kata sandi akun
- Memungkinkan pengguna membuat dan mengambil laporan kredensial IAM
- Izinkan semua tindakan IAM (akses admin)

Memungkinkan pengguna untuk membuat daftar grup akun, pengguna, kebijakan, dan lainnya untuk tujuan pelaporan

Kebijakan berikut memungkinkan pengguna untuk menghubungi setiap tindakan IAM yang dimulai dengan string `Get` atau `List`, dan untuk membuat laporan. Untuk melihat contoh kebijakan, lihat IAM: Mengizinkan akses hanya-baca ke konsol IAM.

Memungkinkan pengguna untuk mengelola keanggotaan grup

Kebijakan berikut memungkinkan pengguna memperbarui keanggotaan grup yang dipanggil MarketingGroup. Untuk melihat contoh kebijakan, lihat IAM: Memungkinkan mengelola keanggotaan grup secara terprogram dan di konsol.

Izinkan pengguna untuk mengelola pengguna IAM

Kebijakan berikut memungkinkan pengguna untuk melakukan semua tugas yang terkait dengan mengelola pengguna IAM tetapi tidak melakukan tindakan pada entitas lain, seperti membuat grup atau kebijakan. Tindakan yang diizinkan meliputi hal berikut ini:

- Membuat pengguna (tindakan `CreateUser`).
- Menghapus pengguna. Tugas ini memerlukan izin untuk melakukan semua tindakan berikut: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup`, dan `DeleteUser`.
- Mencantumkan pengguna dalam akun dan dalam grup (tindakan `GetUser`, `ListUsers`, dan `ListGroupsForUser`).
- Membuat daftar dan menghapus kebijakan untuk pengguna (tindakan `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`)
- Mengganti nama atau mengubah jalur untuk pengguna (tindakan `UpdateUser`). Elemen `Resource` harus menyertakan ARN yang mencakup jalur sumber maupun jalur target. Untuk informasi lebih lanjut tentang jalur, lihat Nama dan jalur yang ramah.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Sejumlah izin yang disertakan dalam kebijakan sebelumnya memungkinkan pengguna melakukan tugas dalam Konsol Manajemen AWS. Pengguna yang melakukan tugas terkait pengguna dari AWS CLI, AWS SDK, atau API kueri HTTP IAM mungkin tidak memerlukan izin tertentu. Misalnya, jika pengguna sudah mengetahui kebijakan ARN untuk melepas dari pengguna, mereka tidak memerlukan izin `iam:ListAttachedUserPolicies`. Daftar pasti izin yang diperlukan pengguna tergantung pada tugas yang harus dilakukan pengguna saat mengelola pengguna lain.

Izin berikut dalam kebijakan ini memungkinkan akses ke tugas pengguna melalui Konsol Manajemen AWS:

- `iam:GetAccount*`
- `iam:ListAccount*`

Izinkan pengguna mengatur kebijakan kata sandi akun

Anda mungkin memberi beberapa pengguna izin untuk mendapatkan dan memperbarui kebijakan ??? kata sandi Anda Akun AWS. Untuk melihat contoh kebijakan, lihat IAM: Memungkinkan pengaturan persyaratan kata sandi akun secara terprogram dan di konsol.

Memungkinkan pengguna membuat dan mengambil laporan kredensial IAM

Anda dapat memberikan izin kepada pengguna untuk membuat dan mengunduh laporan yang mencantumkan semua pengguna di Anda Akun AWS. Laporan tersebut juga mencantumkan status berbagai kredensial pengguna, termasuk sandi, access key, perangkat MFA, dan sertifikat penandatanganan. Untuk informasi lebih lanjut mengenai laporan kredensial, lihat Hasilkan laporan kredensi untuk Anda Akun AWS. Untuk melihat contoh kebijakan, lihat IAM: Menghasilkan dan mengambil laporan kredensi IAM.

Izinkan semua tindakan IAM (akses admin)

Anda dapat memberikan izin administratif kepada beberapa pengguna untuk melakukan semua tindakan di IAM, termasuk mengelola kata sandi, kunci akses, perangkat MFA, dan sertifikat pengguna. Kebijakan contoh berikut memberikan izin ini.

> **Warning**
>
> Saat Anda memberi pengguna akses penuh ke IAM, tidak ada batasan untuk izin yang dapat diberikan pengguna him/herself atau orang lain. Pengguna dapat membuat entitas IAM baru (pengguna atau peran) dan memberikan entitas tersebut akses penuh ke semua sumber daya di Anda Akun AWS. Ketika Anda memberi pengguna akses penuh ke IAM, Anda secara efektif memberi mereka akses penuh ke semua sumber daya di Anda Akun AWS. Ini termasuk akses untuk menghapus semua sumber daya. Anda harus memberikan izin ini hanya kepada administrator tepercaya, dan Anda harus memberlakukan Autentikasi Multi-Faktor (MFA) untuk administrator ini.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Delegasi sementara IAM

Gambaran umum

Delegasi sementara mempercepat orientasi dan menyederhanakan manajemen untuk produk dari Amazon dan AWS Mitra yang terintegrasi dengan AWS akun Anda. Alih-alih mengonfigurasi beberapa AWS layanan secara manual, Anda dapat mendelegasikan izin terbatas sementara yang memungkinkan penyedia produk menyelesaikan tugas penyiapan atas nama Anda dalam hitungan menit melalui alur kerja penerapan otomatis. Anda mempertahankan kontrol administratif dengan persyaratan persetujuan dan batas izin, sementara izin penyedia produk secara otomatis kedaluwarsa setelah durasi yang disetujui tanpa diperlukan pembersihan manual. Jika produk memerlukan akses persisten untuk operasi yang sedang berlangsung, penyedia dapat menggunakan delegasi sementara untuk membuat peran IAM dengan batas izin yang menentukan izin maksimum peran tersebut. Semua aktivitas penyedia produk dilacak AWS CloudTrail untuk kepatuhan dan pemantauan keamanan.

Untuk bekerja dengan permintaan delegasi sementara, Anda dapat memulai permintaan dan men injau permintaan. Untuk informasi selengkapnya tentang pemantauan, lihat notifikasi dan CloudTrail pencatatan.

> **Note**
>
> Permintaan delegasi sementara hanya dapat dibuat oleh produk Amazon dan Mitra yang memenuhi syarat AWS yang telah menyelesaikan proses orientasi fitur. Pelanggan meninjau dan menyetujui permintaan ini tetapi tidak dapat membuatnya secara langsung. Jika Anda

adalah AWS Mitra yang ingin mengintegrasikan delegasi sementara IAM ke dalam produk Anda, lihat Panduan Integrasi Mitra untuk petunjuk orientasi dan integrasi.

Cara kerja delegasi sementara

Delegasi sementara memungkinkan Amazon dan AWS Mitra untuk meminta akses sementara dan terbatas ke akun Anda. Setelah persetujuan Anda, mereka dapat menggunakan izin yang didelegasikan untuk mengambil tindakan atas nama Anda. Permintaan delegasi menentukan izin khusus untuk AWS layanan dan tindakan yang diperlukan penyedia produk untuk menyebarkan atau mengonfigurasi sumber daya di AWS akun Anda. Izin ini hanya tersedia untuk waktu yang terbatas dan secara otomatis kedaluwarsa setelah durasi yang ditentukan dalam permintaan.

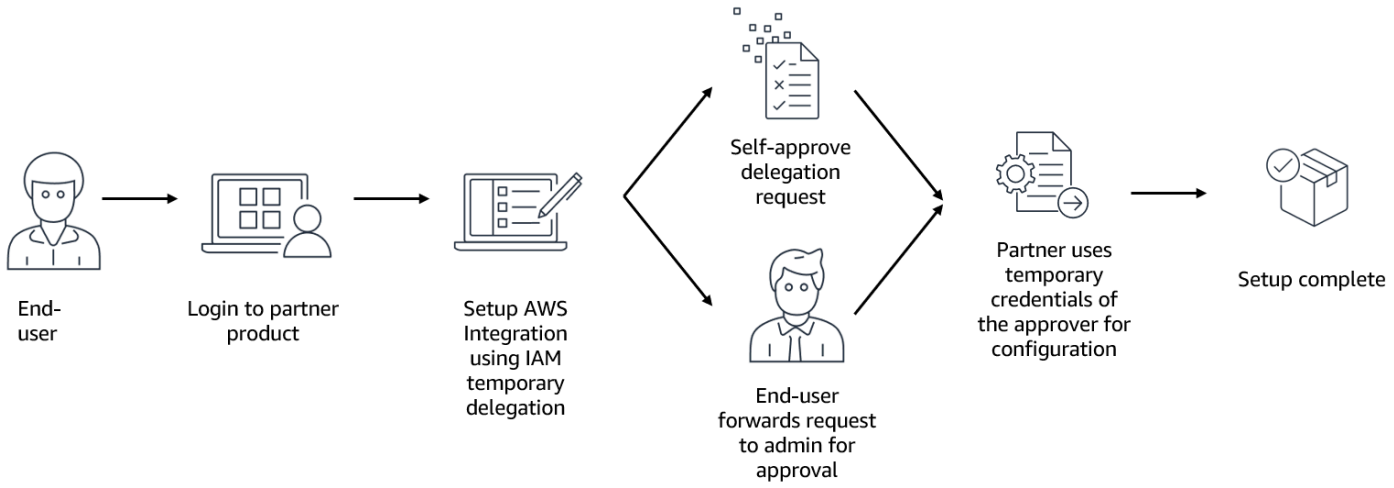
> **Note**
>
> Durasi maksimum untuk akses yang didelegasikan adalah 12 jam. Namun, pengguna root hanya dapat menyetujui permintaan delegasi dengan durasi 4 jam atau kurang. Jika permintaan menentukan lebih dari 4 jam, Anda harus menggunakan identitas non-root untuk menyetujui permintaan. Untuk detailnya, lihat Kemampuan beta simulasi izin.

Untuk tugas yang sedang berlangsung, seperti membaca dari bucket Amazon S3, permintaan delegasi dapat mencakup pembuatan peran IAM yang memungkinkan akses berkelanjutan ke sumber daya dan tindakan setelah akses sementara berakhir. Penyedia produk harus melampirkan batas izin ke setiap peran IAM yang dibuat melalui delegasi sementara. Batas izin membatasi izin maksimum peran tetapi tidak memberikan izin sendiri. Anda dapat meninjau batas izin sebagai bagian dari permintaan sebelum menyetujuinya. Untuk detailnya, lihat Batas izin.

Prosesnya bekerja sebagai berikut:

1. Anda masuk ke produk Amazon atau AWS Mitra untuk mengintegrasikannya dengan AWS lingkungan Anda.
2. Penyedia produk memulai permintaan delegasi atas nama Anda dan mengarahkan Anda ke Konsol AWS Manajemen.
3. Anda meninjau izin yang diminta dan menentukan apakah akan menyetujui, menolak, atau meneruskan permintaan ke administrator Anda.

4. Setelah Anda atau administrator menyetujui permintaan tersebut, penyedia produk dapat memperoleh kredentif sementara pemberi persetujuan untuk melakukan tugas yang diperlukan.
5. Akses penyedia produk secara otomatis kedaluwarsa setelah periode waktu yang ditentukan. Namun, setiap peran IAM yang dibuat melalui permintaan delegasi sementara tetap ada setelah periode ini, memungkinkan penyedia produk untuk terus mengakses sumber daya dan tindakan untuk tugas manajemen yang sedang berlangsung.



> **Note**
>
> Anda hanya dapat mendelegasikan izin ke penyedia produk jika Anda memiliki izin untuk layanan dan tindakan yang disertakan dalam permintaan delegasi sementara. Jika Anda tidak memiliki akses ke layanan dan tindakan yang diminta, penyedia produk tidak menerima izin ini saat Anda menyetujui permintaan.

Jika pemeriksaan izin menunjukkan kemungkinan berhasil, Anda dapat menyetujui permintaan delegasi sementara dan melanjutkan alur kerja.

Jika pemeriksaan izin menunjukkan bahwa Anda mungkin tidak memiliki izin yang memadai, teruskan permintaan ke administrator Anda untuk persetujuan. Sebaiknya beri tahu administrator Anda tentang permintaan ini menggunakan metode pilihan Anda seperti email atau tiket.

Setelah administrator menyetujui permintaan, apa yang terjadi selanjutnya tergantung pada konfigurasi penyedia produk:

- Jika penyedia produk meminta akses segera, mereka secara otomatis menerima izin sementara dan durasi akses dimulai.
- Jika penyedia produk meminta rilis oleh pemilik (penerima awal), Anda harus kembali ke permintaan untuk secara eksplisit membagikan akses akun sementara sebelum durasi akses dimulai. Penyedia produk biasanya menggunakan opsi ini ketika mereka memerlukan masukan tambahan dari Anda, seperti pemilihan sumber daya atau detail konfigurasi, untuk menyelesaikan tugas yang diperlukan.

Memulai permintaan delegasi sementara

Anda dapat memulai permintaan delegasi sementara hanya dari produk Amazon atau AWS Partner yang didukung. Selama alur kerja yang mendukung pendelegasian sementara, Anda akan diminta untuk memberikan izin terbatas sementara kepada penyedia produk untuk mengonfigurasi AWS sumber daya yang diperlukan di akun Anda. Pendekatan otomatis ini memberikan pengalaman yang lebih efisien dengan menghilangkan kebutuhan bagi Anda untuk mengkonfigurasi sumber daya ini secara manual.

Anda dapat meninjau detail permintaan delegasi seperti peran, kebijakan, dan AWS layanan IAM tertentu yang dibutuhkan penyedia produk sebelum memberikan akses. Anda dapat menyetujui permintaan sendiri jika Anda memiliki izin yang memadai, atau meneruskan permintaan tersebut ke administrator akun Anda untuk disetujui. Semua akses penyedia produk dibatasi waktu dan dapat dipantau dan dicabut sesuai kebutuhan.

Untuk memulai permintaan delegasi sementara

1. Arahkan ke konsol produk yang didukung dari Amazon atau AWS Mitra yang memerlukan integrasi dengan AWS akun Anda.
2. Pilih Ter apkan dengan delegasi sementara IAM. Perhatikan bahwa nama opsi mungkin berbeda di seluruh produk yang didukung. Lihat dokumentasi penyedia produk untuk detailnya.

> **Note**
>
> Jika Anda belum masuk ke Konsol AWS Manajemen, jendela baru akan terbuka ke AWS halaman masuk. Sebaiknya Anda masuk ke AWS akun sebelum memulai permintaan delegasi sementara dari konsol produk. Untuk informasi selengkapnya tentang cara masuk berdasarkan jenis pengguna dan AWS sumber daya yang ingin Anda akses, lihat Panduan AWS Sign-In Pengguna.

3. Tinjau detail permintaan untuk mengonfirmasi nama dan AWS akun produk penyedia produk. Anda juga dapat meninjau AWS identitas yang akan digunakan penyedia produk untuk melakukan tindakan atas nama Anda.
4. Tinjau detail Akses untuk izin yang akan didelegasikan sementara dengan menyetujui permintaan ini.

   - Bagian Ringkasan Izin menyediakan ikhtisar tingkat tinggi yang dihasilkan oleh AI yang dapat membantu Anda memahami kategori AWS layanan apa yang dapat diakses dan jenis tindakan apa yang dapat dilakukan di setiap layanan.
   - Pilih Lihat JSON untuk meninjau izin tertentu yang perlu diterapkan penyedia produk di AWS akun Anda, termasuk cakupan akses dan batasan sumber daya.
   - Jika penyedia produk membuat peran IAM sebagai bagian dari permintaan delegasi sementara, batas izin harus dilampirkan ke peran tersebut. Peran IAM ini memiliki izin yang terus mengizinkan akses ke sumber daya dan tindakan setelah durasi akses yang diminta berakhir. Pilih Lihat detail untuk meninjau batas izin, yang menentukan izin maksimum yang dapat dimiliki peran. Penyedia produk akan menerapkan kebijakan tambahan ke peran selama pembuatan yang menentukan izin aktualnya. Kebijakan ini mungkin tampak lebih terfokus, atau lebih luas, daripada batas tergantung pada bagaimana penyedia produk mendefinisikannya. Namun, batas izin menjamin bahwa izin efektif peran tidak akan pernah melebihi apa yang Anda lihat selama persetujuan permintaan, terlepas dari kebijakan apa yang dilampirkan ke peran tersebut. Untuk informasi selengkapnya, lihat Batas izin.
5. Tinjau hasil simulasi izin. Kemampuan simulasi izin secara otomatis mengevaluasi izin identitas Anda terhadap izin yang disertakan dalam permintaan. Berdasarkan analisis ini, rekomendasi ditampilkan yang menunjukkan apakah akan menyetujui permintaan dengan identitas Anda saat ini atau meneruskan permintaan ke administrator. Untuk detailnya, lihat Kemampuan beta simulasi izin.
6. Dalam dialog, pilih bagaimana Anda ingin melanjutkan.

   - Pilih Izinkan akses jika identitas Anda memiliki izin yang cukup untuk mengizinkan penyedia produk melakukan prosedur orientasi atas nama Anda. Bila Anda memilih opsi ini, durasi akses penyedia produk dimulai setelah Anda memberikan akses.
   - Pilih Minta persetujuan jika identitas Anda tidak memiliki izin yang memadai untuk mengizinkan penyedia produk melakukan prosedur orientasi atas nama Anda. Kemudian, pilih Buat permintaan persetujuan. Saat Anda memilih opsi ini, tautan permintaan delegasi sementara dibuat yang dapat Anda bagikan dengan administrator akun Anda. Administrator Anda dapat mengakses Konsol AWS Manajemen atau menggunakan tautan akses untuk

Meninjau permintaan delegasi sementara untuk menyetujui permintaan dan berbagi akses sementara dengan pemohon.

> **Note**
>
> Memberikan akses penyedia produk memerlukan dua tindakan: menerima permintaan delegasi (`AcceptDelegationRequest`) dan melepaskan token pertukaran (`SendDelegationToken`). Konsol AWS Manajemen melakukan kedua langkah secara otomatis saat Anda menyetujui permintaan. Jika Anda menggunakan API AWS CLI atau, Anda harus menjalankan kedua langkah secara terpisah.

Kemampuan simulasi izin -beta

Saat menerima permintaan delegasi sementara, Anda dapat menyetujuinya sendiri atau meneruskannya ke administrator akun Anda untuk disetujui. Anda hanya dapat mendelegasikan izin ke penyedia produk jika Anda memiliki izin untuk layanan dan tindakan yang disertakan dalam permintaan delegasi sementara. Jika Anda tidak memiliki akses ke layanan dan tindakan yang diminta, penyedia produk tidak akan menerima izin tersebut meskipun izin tersebut disertakan dalam permintaan.

Misalnya, permintaan delegasi sementara memerlukan kemampuan untuk membuat bucket Amazon S3, memulai dan menghentikan instans di Amazon EC2, dan mengambil peran IAM. Identitas yang menyetujui permintaan dapat memulai dan menghentikan instans di Amazon EC2, dan mengambil peran IAM, tetapi tidak memiliki izin untuk membuat bucket Amazon S3. Ketika identitas ini menyetujui permintaan, penyedia produk tidak dapat membuat bucket Amazon S3 meskipun izin ini disertakan dalam permintaan delegasi sementara.

Karena Anda hanya dapat mendelegasikan izin yang sudah Anda miliki, penting untuk mengevaluasi apakah Anda memiliki izin yang diminta sebelum menyetujui. Kemampuan simulasi izin beta membantu evaluasi ini dengan membandingkan izin Anda dengan izin yang disertakan dalam permintaan. Penilaian menunjukkan apakah Anda dapat menyetujui permintaan dengan identitas Anda saat ini atau perlu meneruskannya ke administrator. Jika analisis tidak dapat memvalidasi bahwa Anda memiliki izin yang memadai, teruskan permintaan ke administrator untuk ditinjau. Penilaian ini didasarkan pada analisis izin simulasi dan mungkin berbeda dari AWS lingkungan langsung Anda, jadi tinjau izin yang diminta dengan cermat sebelum melanjutkan.

Langkah selanjutnya

Setelah memulai permintaan delegasi sementara, Anda dapat mengelola dan memantau permintaan melalui siklus hidupnya. Prosedur berikut membantu Anda melacak, menyetujui, dan mengontrol akses sementara:

- Meninjau permintaan delegasi sementara — Pantau status permintaan akses Anda, dan lihat informasi terperinci tentang permintaan untuk menyetujui atau menolak permintaan delegasi sementara.
- Cabut akses delegasi sementara — Segera hentikan sesi delegasi sementara yang aktif sebelum kedaluwarsa secara alami.

Tinjau permintaan delegasi sementara

Setelah memulai permintaan delegasi sementara, Anda dapat memantau, menyetujui, dan menolak permintaan di konsol IAM. Halaman Permintaan delegasi Sementara menyediakan tampilan terpusat dari semua permintaan, termasuk permintaan yang menunggu persetujuan, selesai, atau ditolak. Sebagai administrator, Anda dapat meninjau permintaan ini untuk memberikan akses kepada penyedia produk ke AWS sumber daya atau menolaknya berdasarkan kebijakan keamanan, persyaratan bisnis, atau standar kepatuhan organisasi Anda. Visibilitas ini membantu Anda melacak siklus hidup akses penyedia produk dan menjaga pengawasan izin sementara.

> **Note**
>
> Anda harus memiliki AcceptDelegationRequest izin iam: untuk menyetujui permintaan delegasi sementara.

Untuk menyetujui permintaan delegasi sementara

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Permintaan delegasi sementara.
3. Halaman utama menampilkan daftar permintaan delegasi sementara Anda dengan informasi berikut:
    - ID Permintaan - Pengidentifikasi unik untuk permintaan

- Status - Status saat ini (Tertunda, Disetujui, Ditolak, Dibagikan, Kedaluwarsa)
- Pemohon - Penyedia produk yang terkait dengan permintaan
- Diprakarsai oleh - kepala IAM di akun yang memprakarsai permintaan untuk penyedia produk
- Permintaan dibuat - Saat permintaan dikirimkan
- Permintaan kedaluwarsa — Ketika permintaan kedaluwarsa atau akan kedaluwarsa

4. (Opsional) Gunakan opsi filter untuk melihat permintaan berdasarkan status:

   - Semua permintaan — Lihat semua permintaan Anda terlepas dari status
   - Tertunda - Lihat permintaan menunggu persetujuan administrator
   - Disetujui - Lihat permintaan yang disetujui
   - Dibagikan - Lihat permintaan yang aksesnya telah dibagikan
   - Ditolak - Lihat permintaan yang ditolak dengan alasan penolakan

5. Untuk melihat informasi terperinci tentang permintaan tertentu atau meninjau permintaan persetujuan yang tertunda, pilih ID permintaan.
6. Tinjau informasi permintaan terperinci:

   - Informasi penyedia produk
   - Minta alasan dan pembenaran
   - Durasi yang diminta
   - AWS Izin yang diminta

7. Jika Anda seorang administrator yang meninjau permintaan yang tertunda, pilih salah satu opsi berikut:

   - Untuk menyetujui permintaan, pilih Menyetujui. Dalam dialog persetujuan, Anda dapat melihat hasil simulasi izin. Untuk informasi selengkapnya, lihat Kemampuan beta simulasi izin. Setelah mengonfirmasi durasi akses dan AWS identitas Anda, pilih Menyetujui untuk memberikan akses. Jika penyedia produk meminta akses langsung, mereka secara otomatis menerima izin sementara dan durasi akses dimulai. Jika tidak, beri tahu orang yang memulai permintaan untuk melepaskan akses ke penyedia produk.
   - Untuk menolak permintaan, pilih Tolak.
     - Dalam dialog penolakan, berikan alasan yang jelas untuk penolakan untuk membantu pemohon memahami mengapa permintaan mereka ditolak.
     - Pilih Tolak untuk menolak akses.

8. Daftar permintaan secara otomatis menyegarkan untuk menampilkan informasi status terbaru. Anda juga dapat menyegarkan halaman secara manual untuk memeriksa pembaruan status.

Cabut akses delegasi sementara

Meskipun sesi akses penyedia produk dirancang untuk kedaluwarsa secara otomatis setelah durasi yang disetujui, Anda mungkin perlu segera menghentikan akses dalam situasi tertentu. Pencabutan akses penyedia produk aktif menyediakan mekanisme kontrol darurat ketika masalah keamanan muncul, ketika pekerjaan penyedia produk selesai lebih awal, atau ketika persyaratan bisnis berubah. Baik inisiator permintaan maupun administrator dapat mencabut akses untuk menjaga keamanan dan kontrol operasional.

Untuk mencabut akses delegasi sementara

1. Masuk ke Konsol AWS Manajemen dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Per mintaan delegasi sementara.
3. Cari ID permintaan untuk sesi akses yang ingin Anda cabut.
4. Pilih Tind akan lalu pilih Cabut akses.
5. Dalam dialog, pilih Cab ut akses untuk mengonfirmasi bahwa Anda ingin segera menghentikan sesi akses.

Setelah mencabut akses, penyedia produk tidak akan lagi dapat mengakses AWS sumber daya Anda. Pencabutan masuk AWS CloudTrail untuk tujuan audit.

> **Important**
>
> Pencabutan akses segera mengakhiri sesi akses penyedia produk. Setiap pekerjaan atau proses yang sedang berlangsung menggunakan akses akan terganggu. Pastikan bahwa pencabutan tidak akan mengganggu operasi kritis.

> **Note**
>
> Anda tidak dapat mencabut akses untuk permintaan yang disetujui menggunakan pengguna root. AWS menyarankan agar Anda menghindari menggunakan pengguna root untuk

menyetujui permintaan delegasi. Gunakan peran IAM dengan izin yang sesuai sebagai gantinya.

Mengelola Izin untuk Permintaan Delegasi

Administrator dapat memberikan izin kepada prinsipal IAM untuk mengelola permintaan delegasi dari penyedia produk. Ini berguna ketika Anda ingin mendelegasikan otoritas persetujuan ke pengguna atau tim tertentu di organisasi Anda, atau ketika Anda perlu mengontrol siapa yang dapat melakukan tindakan tertentu pada permintaan delegasi.

Izin IAM berikut tersedia untuk mengelola permintaan delegasi:

Izin	Deskripsi
sudah: AssociateDelegationRequest	Mengaitkan permintaan delegasi yang tidak ditugaskan dengan akun Anda AWS
sudah: GetDelegationRequest	Melihat detail permintaan delegasi
sudah: UpdateDelegationRequest	Meneruskan permintaan delegasi ke administrator untuk disetujui
sudah: AcceptDelegationRequest	Menyetujui permintaan delegasi
sudah: SendDelegationToken	Lepaskan token pertukaran ke penyedia produk setelah persetujuan
sudah: RejectDelegationRequest	Menolak permintaan delegasi
sudah: ListDelegationRequests	Daftar permintaan delegasi untuk akun Anda

> **Note**
>
> Secara default, prinsipal IAM yang memulai permintaan delegasi secara otomatis diberikan izin untuk mengelola permintaan spesifik tersebut. Mereka dapat mengaitkannya dengan akun mereka, melihat detail permintaan, menolak permintaan, meneruskannya ke administrator untuk persetujuan, melepaskan token pertukaran ke penyedia produk setelah persetujuan admin, dan mencantumkan permintaan delegasi yang mereka miliki.

Notifikasi

Delegasi sementara IAM terintegrasi dengan Pemberitahuan AWS Pengguna untuk membantu Anda tetap mendapat informasi tentang perubahan status permintaan delegasi. Pemberitahuan sangat berguna bagi administrator yang perlu meninjau dan menyetujui permintaan delegasi.

Dengan Pemberitahuan AWS Pengguna, Anda dapat mengonfigurasi peringatan untuk dikirimkan melalui beberapa saluran, termasuk email, Amazon Simple Notification Service (SNS), AWS Chatbot untuk Slack atau Microsoft Teams, dan Console Mobile Application. AWS Ini memastikan bahwa orang yang tepat diberi tahu pada waktu yang tepat, memungkinkan respons yang lebih cepat terhadap persetujuan yang tertunda atau kesadaran akan perubahan akses. Anda juga dapat menyesuaikan peristiwa mana yang memicu pemberitahuan berdasarkan kebutuhan organisasi dan persyaratan keamanan.

Acara Pemberitahuan yang Tersedia

Anda dapat berlangganan untuk menerima pemberitahuan untuk acara delegasi sementara IAM berikut:

- Permintaan Delegasi Sementara IAM Dibuat
- Permintaan Delegasi Sementara IAM Ditugaskan
- Permintaan Delegasi Sementara IAM Menunggu Persetujuan
- Permintaan Delegasi Sementara IAM Ditolak
- Permintaan Delegasi Sementara IAM Diterima
- Permintaan Delegasi Sementara IAM Diselesaikan
- Permintaan Delegasi Sementara IAM Kedaluwarsa

Mengkonfigurasi Pemberitahuan

Untuk mengonfigurasi pemberitahuan untuk acara delegasi sementara IAM:

1. Buka konsol Pemberitahuan AWS Pengguna
2. Membuat atau memperbarui konfigurasi notifikasi
3. Pilih AWS IAM sebagai layanan
4. Pilih acara permintaan delegasi mana yang ingin Anda beri tahu
5. Konfigurasikan saluran pengiriman Anda (email, AWS Chatbot, dll.)

Untuk petunjuk mendetail tentang mengonfigurasi Pemberitahuan AWS Pengguna, termasuk menyiapkan saluran pengiriman dan mengelola aturan notifikasi, lihat dokumentasi Pemberitahuan AWS Pengguna.

CloudTrail

Semua tindakan yang dilakukan oleh penyedia produk menggunakan akses delegasi sementara secara otomatis masuk AWS CloudTrail. Ini memberikan visibilitas lengkap dan auditabilitas aktivitas penyedia produk di AWS akun Anda. Anda dapat mengidentifikasi tindakan yang diambil oleh penyedia produk, kapan tindakan tersebut terjadi, dan akun penyedia produk mana yang melakukannya.

Untuk membantu Anda membedakan antara tindakan yang diambil oleh prinsipal IAM Anda sendiri dan tindakan yang diambil oleh penyedia produk dengan akses yang didelegasikan, CloudTrail acara menyertakan bidang baru yang disebut `invokedByDelegate` di bawah elemen. `userIdentity` Bidang ini berisi ID AWS akun penyedia produk, sehingga memudahkan untuk memfilter dan mengaudit semua tindakan yang didelegasikan.

CloudTrail Struktur Acara

Contoh berikut menunjukkan CloudTrail peristiwa untuk tindakan yang dilakukan oleh penyedia produk menggunakan akses yang didelegasikan sementara:

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "[REDACTED:AWS_ACCESS_KEY]",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedByDelegate": {
            "accountId": "444455556666"
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePolicyName"
    },
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


Bid `invokedByDelegate` ang berisi ID AWS akun penyedia produk yang melakukan tindakan menggunakan akses yang didelegasikan. Dalam contoh ini, akun 444455556666 (penyedia produk) melakukan tindakan di akun 111122223333 (akun pelanggan).

Delegasi sementara IAM untuk AWS Partner

Gambaran umum

Delegasi sementara IAM memungkinkan AWS pelanggan untuk secara mulus and/or mengintegrasikan produk AWS Mitra ke AWS lingkungan mereka melalui alur kerja yang interaktif

dan terpandu. Pelanggan dapat AWS memberikan Mitra akses sementara terbatas untuk mengonfigurasi AWS layanan yang diperlukan, mengurangi gesekan orientasi dan mempercepat waktu menuju nilai.

Untuk mengintegrasikan delegasi sementara ke dalam produk Anda, pelajari tentang memahami integrasi Anda, tin jau izin pemahaman, dan ikuti pedoman evaluasi kebijakan. Untuk contoh kebijakan IAM, lihat templ at kebijakan. Ketika Anda siap untuk mengimplementasikan, lihat membangun integrasi Anda.

Delegasi sementara IAM memungkinkan mitra untuk:

- Merampingkan orientasi pelanggan dengan penyediaan sumber daya otomatis
- Mengurangi kompleksitas integrasi dengan menghilangkan langkah-langkah konfigurasi manual
- Bangun kepercayaan melalui izin yang transparan dan disetujui pelanggan
- Aktifkan operasi yang sedang berlangsung dengan pola akses jangka panjang menggunakan batas izin

Cara kerjanya

1. Mitra membuat permintaan delegasi - Mitra membuat permintaan yang menentukan izin apa yang mereka butuhkan dan untuk berapa lama
2. Ulasan pelanggan di AWS Konsol - Pelanggan melihat dengan tepat izin apa yang diminta mitra dan alasannya
3. Pelanggan menyetujui - Pelanggan menyetujui permintaan dan melepaskan token pertukaran. Token dikirim ke mitra pada topik SNS yang ditentukan ini.
4. Mitra menerima kredenSIAL sementara - Mitra menukar token dengan AWS kredenSIAL sementara
5. Mitra mengonfigurasi sumber daya - Mitra menggunakan kredensional untuk mengatur sumber daya yang diperlukan di akun pelanggan

Kualifikasi mitra

Agar memenuhi syarat untuk integrasi delegasi sementara, mitra harus memenuhi persyaratan berikut:

- Partisipasi ISV Acceler ate — Anda harus terdaftar dalam program ISV Accelerate (ISVA).

- AWS Daftar Marketplace — Produk Anda harus terdaftar di AWS Marketplace dengan lencana “Diterapkan AWS pada”.

Proses onboarding

Selesaikan langkah-langkah berikut untuk mengintegrasikan delegasi sementara ke dalam produk Anda:

1. Langkah 1: Tinjau persyaratan

   Tinjau dokumentasi ini untuk memahami persyaratan kualifikasi dan lengkapi kuesioner mitra di bawah ini.

2. Langkah 2: Kirim permintaan onboarding Anda

   Kirim email ke aws-iam-partner-onboarding@amazon.com atau hubungi AWS perwakilan Anda. Sertakan kuesioner mitra Anda yang telah diisi dengan semua bidang wajib dari tabel di bawah ini.

3. Langkah 3: AWS validasi dan peninjauan

   AWS akan:

   - Memvalidasi bahwa Anda memenuhi kriteria kualifikasi
   - Meninjau templat kebijakan dan batas izin Anda
   - Berikan umpan balik tentang artefak yang Anda kirimkan

4. Langkah 4: Perbaiki kebijakan Anda

   Tanggapi AWS umpan balik dan kirimkan templat kebijakan atau batas izin yang diperbarui sesuai kebutuhan.

5. Langkah 5: Selesaikan pendaftaran

   Setelah persetujuan, AWS akan:

   - Aktifkan akses API untuk akun yang Anda tentukan
   - Bagikan ARN untuk templat kebijakan dan batas izin Anda (jika berlaku)

   Anda akan menerima konfirmasi saat onboarding selesai. Anda kemudian dapat mengakses API delegasi sementara, CreateDelegationRequest dan GetDelegatedAccessToken dari akun

terdaftar Anda dan mulai mengintegrasikan alur kerja permintaan delegasi ke dalam produk Anda.

Kuesioner mitra

Tabel berikut mencantumkan informasi yang diperlukan untuk orientasi mitra:

Informasi	Deskripsi	Diperlukan
ID Akun Pusat Mitra	ID akun AWS akun terdaftar Anda di AWS Partner Central.	Ya
PartnerId	ID Mitra disediakan oleh AWS Partner Central.	Tidak
AWS Id Produk Pasar	ID Produk untuk produk Anda disediakan oleh AWS Partner Central.	Ya
AWS AkunTid	Daftar ID AWS Akun yang ingin Anda gunakan untuk memanggil API delegasi sementara. Ini harus mencakup produksi dan non- production/test akun Anda.	Ya
Nama mitra	Nama ini ditampilkan kepada pelanggan di Konsol AWS Manajemen saat mereka meninjau permintaan delegasi sementara Anda.	Ya
Email kontak	Satu atau lebih alamat email untuk menghubungi Anda tentang integrasi Anda.	Ya
Domain Pemohon	Domain Anda (misalnya, www.example.com)	Ya
Deskripsi integrasi	Deskripsi singkat tentang kasus penggunaan yang ingin Anda atasi menggunakan fitur ini. Anda dapat menyertakan tautan referensi ke dokumentasi Anda atau materi publik lainnya.	Ya
Diagram arsitektur	Diagram arsitektur yang menggambarkan kasus penggunaan integrasi Anda.	Tidak

Informasi	Deskripsi	Diperlukan
Template kebijakan	Anda harus mendaftarkan setidaknya satu templat kebijakan untuk fitur ini. Template kebijakan menentuka n izin sementara yang ingin Anda minta di AWS akun pelanggan. Untuk informasi selengkapnya, lihat bagian Template kebijakan.	Ya
Nama templat kebijakan	Nama template kebijakan yang ingin Anda daftarkan.	Ya
Batas Izin	Jika Anda ingin membuat peran IAM di akun pelanggan menggunakan izin sementara, Anda harus mendaftar kan batas izin dengan IAM. Batas izin akan dilampirk an ke peran IAM yang Anda buat untuk membatasi izin maksimum pada peran. Anda dapat menggunakan kebijakan ter AWS kelola yang dipilih sebagai batas izin atau mendaftarkan batas izin khusus (JSON) baru. Untuk informasi selengkapnya, lihat bagian Batas Izin.	Tidak
Nama Batas Izin	Nama batas izin Anda. Formatnya adalah: arn:aws:i am: :partner:policy/permission_boundary/<partner_ domain>/<policy_name>_ <date>Nama kebijakan harus menyertakan tanggal pembuatan sebagai akhiran. Nama tidak dapat diperbarui setelah batas izin dibuat. Jika Anda menggunakan kebijakan terkelola yang AWS ada, berikan kebijakan terkelola ARN sebagai gantinya.	Tidak
Deskripsi Batas Izin	Deskripsi untuk batas izin. Deskripsi ini tidak dapat diperbarui setelah batas izin dibuat.	Tidak

Memahami integrasi Anda

Setelah menyelesaikan proses orientasi, Anda dapat membangun integrasi dengan delegasi sementara IAM. Integrasi lengkap biasanya melibatkan tiga kategori utama pekerjaan:

1. Pengalaman Pengguna dan Desain Alur Kerja

Bangun pengalaman front-end di aplikasi mitra yang memandu pelanggan melalui alur kerja delegasi sementara. Aplikasi mitra harus:

- Sajikan alur orientasi atau konfigurasi yang jelas di mana pelanggan dapat memberikan akses sementara. Beri label tindakan ini dengan jelas, seperti “Terapkan dengan delegasi sementara IAM”.
- Mengalihkan pelanggan ke AWS Management Console untuk meninjau dan menyetujui permintaan delegasi menggunakan tautan konsol yang dikembalikan oleh CreateDelegationRequest API
- Berikan pesan yang sesuai tentang izin apa yang diminta dan mengapa. Pelanggan dapat melihat pesan ini di halaman detail permintaan delegasi.
- Tangani pengembalian pelanggan ke aplikasi Anda setelah mereka menyelesaikan persetujuan AWS.

Praktik Terbaik untuk Permintaan Delegasi Sementara

Saat menerapkan delegasi sementara untuk aplikasi mitra Anda, ikuti praktik terbaik yang ditetapkan di bawah ini untuk membantu pelanggan memverifikasi keaslian dan akurasi permintaan.

1. Sertakan User-Identifiable Konteks dalam Pesan Permintaan

Anda harus menyertakan konteks khusus pengguna dalam pesan permintaan delegasi. Informasi ini membantu pengguna mengidentifikasi permintaan dari alur kerja mitra mereka yang ada dan membedakan permintaan yang sah. Informasi yang disarankan dalam pesan permintaan:

- Pengidentifikasi akun atau nama pengguna pelanggan di layanan Anda
- Nama ruang kerja, ID langganan, atau pengidentifikasi organisasi
- Sumber daya spesifik yang diakses (nama cluster, nama proyek, lingkungan)
- Pengidentifikasi transaksi atau permintaan unik yang dihasilkan untuk upaya delegasi ini

Contoh pesan permintaan:

```
Request from Partner A workspace "production-analytics"
Account: user@example.com
```

```
Workspace ID: 1234ABCD
Cluster: ml-training-cluster-01
Request ID: 1111-2222-3333-4444
```

2. Opsional: Sertakan ID AWS Akun saat Memulai Delegasi

Jika ID akun pelanggan tersedia, sertakan dalam permintaan delegasi. Langkah verifikasi ini menciptakan ikatan tambahan antara niat pelanggan dan token delegasi.

3. Pesan Permintaan Desain untuk Verifikasi Keamanan

Struktur pesan permintaan Anda untuk memungkinkan pelanggan memverifikasi legitimasi dengan percaya diri sebelum memberikan akses. Persyaratan pengalaman pengguna:

- Menampilkan pesan permintaan secara mencolok di antarmuka aplikasi Anda sebelum mengalihkan ke AWS
- Gunakan bahasa yang jelas dan deskriptif yang terhubung langsung ke tindakan pelanggan saat ini
- Hindari pesan generik yang dapat diterapkan pada permintaan delegasi apa pun
- Sertakan detail yang cukup bagi pelanggan untuk memverifikasi permintaan cocok dengan alur kerja yang mereka inginkan
- Menampilkan ID AWS akun yang akan menerima akses yang didelegasikan
- Berikan penjelasan yang jelas tentang izin apa yang akan diminta aplikasi Anda

Sertakan pemberitahuan konfirmasi dalam aplikasi Anda sebelum mengarahkan pelanggan ke layar AWS persetujuan:

```
## Redirecting to IAM Temporary Delegation
You are about to grant [Your Service Name] temporary access to your AWS account.

Before clicking "Allow" on the AWS consent screen:
• Verify the request details match your current action
• Confirm the AWS account ID matches your intended account
• Make sure you initiated this request from [Your Service Name]
```

4. Rekomendasi Pengikatan Sesi

Sementara AWS IAM menangani alur otorisasi inti, terapkan praktik ini di aplikasi Anda untuk memperkuat integritas sesi:

- Hasilkan pengidentifikasi permintaan sekali pakai yang unik untuk setiap upaya delegasi
- Mengaitkan permintaan delegasi dengan sesi aplikasi aktif pelanggan
- Menerapkan kedaluwarsa permintaan yang sesuai tergantung pada proses delegasi Anda
- Validasi bahwa panggilan balik delegasi cocok dengan konteks permintaan asal
- Catat semua inisiasi dan penyelesaian permintaan delegasi untuk pemantauan keamanan

Dengan mengikuti praktik terbaik ini, Anda membantu menciptakan pengalaman delegasi sementara yang lebih aman.

2. Integrasi API

Gunakan API delegasi sementara IAM untuk mengirim dan mengelola permintaan delegasi. Setelah AWS akun Anda terdaftar, Anda dapat mengakses API berikut:

- IAM CreateDelegationRequest — Membuat permintaan delegasi untuk AWS akun pelanggan. API ini mengembalikan tautan konsol tempat Anda mengarahkan pelanggan untuk meninjau dan menyetujui permintaan.
- AWS STS GetDelegatedAccessToken— Mengambil kre AWS dentif sementara setelah pelanggan menyetujui permintaan delegasi Anda. Gunakan kredenSIAL ini untuk melakukan tindakan di akun pelanggan.

Integrasi Anda harus menangani siklus hidup lengkap permintaan delegasi, termasuk membuat permintaan, memantau statusnya, dan mengambil kredentif sementara saat disetujui.

3. Konfigurasi dan Orkestrasi Sumber Daya

Setelah mendapatkan kredenSIAL sementara, atur alur kerja yang diperlukan untuk mengonfigurasi sumber daya di akun pelanggan AWS . Ini mungkin termasuk:

- Memanggil API AWS layanan secara langsung untuk membuat dan mengonfigurasi sumber daya
- Menyebarkan infrastruktur menggunakan templat AWS CloudFormation
- Membuat peran IAM untuk akses berkelanjutan (memerlukan penggunaan batas izin). Sebagai ukuran pertahanan mendalam, sebaiknya gunakan nama peran dinamis, misalnya, nama dengan

akhiran yang dibuat secara acak. Pendekatan ini membuat nama peran di setiap akun pelanggan unik dan tidak dapat diprediksi.

Logika orkestrasi Anda harus idempoten dan menangani kegagalan dengan anggun, karena pelanggan mungkin perlu mencoba lagi atau memodifikasi persetujuan delegasi mereka.

Memahami izin

Sebagai bagian dari proses orientasi fitur, Anda harus mendaftarkan kebijakan dengan IAM yang menentukan izin yang ingin Anda minta di akun pelanggan AWS . Proses pendaftaran memberikan pengalaman yang lebih konsisten bagi pelanggan dan membantu menghindari jebakan umum dalam pembuatan kebijakan.

Selama pendaftaran, AWS evaluasi kebijakan Anda terhadap serangkaian validasi. Validasi ini dimaksudkan untuk menstandarkan pemformatan dan struktur kebijakan, dan memberikan perlindungan dasar terhadap anti-pola yang diketahui. Validasi juga mengurangi risiko eskalasi hak istimewa, akses lintas akun yang tidak diinginkan, dan akses luas ke sumber daya bernilai tinggi di akun pelanggan.

Jenis Izin

AWS akan mempertimbangkan dua kategori izin: sementara dan jangka panjang.

Izin Sementara

Izin sementara membatasi izin yang ditetapkan ke sesi akses sementara yang didelegasikan. Izin sementara dijelaskan dalam templat kebijakan yang diterapkan ke sesi yang didelegasikan. Template mendukung parameter yang Anda berikan saat membuat permintaan delegasi. Nilai parameter ini kemudian terikat ke sesi. Izin sementara bekerja dengan cara yang sama seperti kebijakan sesi yang tersedia AWS STS saat ini: kebijakan membatasi kemampuan pengguna yang mendasarinya, tetapi tidak memberikan akses tambahan apa pun. Untuk informasi selengkapnya, lihat AWS STS dokumentasi tentang kebijakan sesi.

Long-Term Izin

Long-term izin membatasi izin peran apa pun yang dibuat atau dikelola melalui akses sementara. Long-term izin diimplementasikan sebagai batas izin IAM. Anda dapat mengirimkan satu atau lebih batas izin AWS sebagai bagian dari orientasi. Setelah disetujui, AWS akan membagikan kebijakan ARN dengan Anda yang dapat Anda rujuk dalam kebijakan Anda.

Kebijakan batas ini memiliki dua fitur penting. Pertama, mereka tidak berubah. Jika Anda ingin memperbarui izin, Anda dapat mendaftarkan batas izin baru. Anda kemudian dapat melampirkan batas izin baru ke peran pelanggan Anda dengan mengirimkan permintaan delegasi baru. Kedua, kebijakan tidak diberi templat. Karena kebijakan batas yang sama dibagikan secara global, kebijakan tersebut tidak dapat diubah berdasarkan per pelanggan.

> **Important**
>
> Batas izin memiliki batas ukuran maksimum 6.144 karakter.

> **Note**
>
> Jika Anda ingin memperbarui batas izin atau templat kebijakan, hubungi IAM di aws-iam-partner-onboarding@amazon.com. Setelah batas izin baru terdaftar, Anda kemudian dapat mengirim permintaan delegasi ke pelanggan untuk memperbarui peran IAM dan melampirkan batas izin yang baru terdaftar. Lihat bagian Contoh untuk lebih jelasnya.

Contoh Kasus Penggunaan: Beban Kerja Pemrosesan Data

Pertimbangkan penyedia produk yang menjalankan beban kerja pemrosesan data di akun pelanggan. Penyedia perlu menyiapkan infrastruktur selama orientasi awal, tetapi juga memerlukan akses berkelanjutan untuk mengoperasikan beban kerja.

Izin sementara (untuk pengaturan awal):

- Membuat instans Amazon EC2, VPC, dan grup keamanan
- Membuat bucket Amazon S3 untuk data yang diproses
- Membuat peran IAM untuk operasi yang sedang berlangsung
- Melampirkan batas izin ke peran IAM

Long-term izin (peran IAM dengan batas izin untuk operasi yang sedang berlangsung):

- Mulai dan hentikan instans Amazon EC2 untuk menjalankan pekerjaan pemrosesan
- Membaca data input dari bucket Amazon S3
- Tulis hasil yang diproses ke bucket Amazon S3

Izin sementara digunakan sekali selama onboarding untuk mengkonfigurasi infrastruktur. Peran IAM yang dibuat selama proses ini memiliki batas izin yang membatasi izin maksimumnya hanya untuk operasi yang diperlukan untuk manajemen beban kerja yang sedang berlangsung. Ini memastikan bahwa meskipun kebijakan peran diubah, itu tidak dapat melebihi izin yang ditentukan dalam batas.

Pedoman evaluasi kebijakan

AWS akan mengevaluasi kebijakan yang Anda kirimkan terhadap serangkaian pedoman. Pedoman evaluasi yang sama berlaku untuk templat kebijakan dan batas izin, dengan perbedaan kecil dicatat jika sesuai.

Untuk tujuan evaluasi, layanan dipisahkan menjadi kelompok-kelompok yang berbeda. Perbedaan yang paling penting adalah untuk layanan sensitif keamanan, yang mengelola akses, kredenSIAL, dan kunci. Kebijakan yang memberikan akses ke layanan ini perlu difokuskan secara sempit pada pekerjaan yang sedang dilakukan. Security-sensitive layanan meliputi: AWS Identitas dan Manajemen Akses (IAM), Layanan Manajemen K AWS unci (KMS), Manajer Akses Sumber AWS Daya (RAM), Pusat Identitas AWS IAM, AWS Organisasi, dan AWS Manajer Rahasia.

Perbedaan sekunder adalah layanan yang dapat mengakses data melintasi batas akun. Kebijakan untuk layanan ini harus mencakup perlindungan untuk mencegah akses lintas akun yang tidak disengaja.

Validasi umum

Semua pernyataan kebijakan harus mengikuti pedoman berikut:

- Semua pernyataan harus menyertakan bidang Efek, Tindakan (atau NotAction), Sumber Daya, dan Kondisi dalam urutan tersebut
- Semua tindakan dalam satu pernyataan harus dicantumkan menurut abjad
- Semua ARN yang disertakan dalam kebijakan harus mengikuti sintaks yang ditentukan dalam dokumentasi publik untuk layanan yang relevan
- NotAction bidang hanya dapat digunakan dalam pernyataan Deny
- Tindakan dalam pernyataan Izinkan harus menyertakan kode layanan. Karakter pengganti generik (“*”) tidak diperbolehkan

Security-sensitive pembatasan layanan

Pembatasan berikut berlaku untuk layanan sensitif keamanan yang disebutkan di atas:

- Tindakan dalam pernyataan Izinkan harus lebih spesifik daripada [service] :*
- Tindakan dalam Pernyataan Izinkan untuk templat kebijakan akses sementara tidak boleh berisi wildcard
- Tindakan sensitif, seperti iam: PassRole atau iam:CreateServiceLinkedRole, memerlukan cakupan tambahan, seperti sumber daya tertentu atau pemeriksaan bersyarat. Tindakan ini meliputi:
  - Penerusan peran IAM
  - Tindakan modifikasi peran IAM
  - Tindakan modifikasi kebijakan IAM
  - AWS Penulisan KMS atau operasi kriptografi
  - AWS Operasi menulis atau berbagi RAM
  - AWS Operasi Secrets Manager untuk mengambil atau memodifikasi rahasia, atau memodifikasi kebijakan sumber daya
- Tindakan lain mungkin menggunakan sumber daya wildcard, seperti iam: ListUsers atau iam: GetPolicy
- Tindakan yang mengelola kredenSIAL, seperti iam:CreateAccessKey, diblokir

IAM-specific pembatasan

Untuk IAM:

- Hanya operasi penulisan terbatas yang diizinkan untuk peran dan kebijakan IAM. Anda tidak dapat meminta izin pada sumber daya IAM lainnya seperti pengguna, grup, dan sertifikat.
- Lampiran kebijakan atau tindakan manajemen kebijakan sebaris terbatas pada peran dengan batas izin. Batas izin harus disediakan oleh mitra atau dalam daftar kebijakan terkelola yang diizinkan AWS . AWS kebijakan yang dikelola mungkin diizinkan jika mereka tidak memberikan hak istimewa atau izin administratif yang tinggi. Misalnya, kebijakan AWS terkelola untuk fungsi pekerjaan tertentu atau SecurityAudit kebijakan mungkin dapat diterima. AWS akan meninjau setiap kebijakan yang AWS dikelola berdasarkan kasus per kasus selama proses orientasi.
- Manajemen kebijakan hanya diperbolehkan untuk kebijakan dengan jalur khusus mitra: arn:aws:iam: :@ {}: .com/ [feature] * AccountId policy/partner_domain
- Tag mungkin hanya diterapkan selama pembuatan sumber daya, dan hanya untuk peran dan kebijakan
- iam: PassRole cek harus cocok dengan nama atau awalan jalur tertentu

AWS STS-pembatasan khusus

Untuk AWS STS:

- sts: AssumeRole harus dicakup ke peran tertentu ARN, awalan ARN peran, atau terbatas pada satu set akun atau unit organisasi ID/organizational

Pembatasan Pusat Identitas IAM

Untuk Pusat Identitas AWS IAM, tindakan berikut diblokir:

- Semua tindakan yang berhubungan dengan manajemen izin (misalnya, sso:AttachCustomerManagedPolicyReferenceToPermissionSet)
- Modifikasi pengguna, grup, dan keanggotaan untuk AWS Identity Store
- Manajemen tag

AWS Pembatasan organisasi

Untuk AWS Organisasi, hanya tindakan baca yang diizinkan.

Validasi khusus layanan tambahan

- Tindakan yang memperoleh rahasia atau kredenSIAL, seperti glue: GetConnection atau redshift:GetClusterCredentials, harus memiliki kondisi yang cocok dengan ARN penuh, awalan ARN, atau tag
- Untuk Amazon Redshift: redshift: hanya GetClusterCredentials diperbolehkan pada nama database tertentu, dan redshift: GetClusterCredentialsWithIAM hanya diperbolehkan pada nama grup kerja tertentu

> **Note**
>
> Saat mengelola sumber daya IAM di akun, sebaiknya gunakan jalur yang menunjukkan nama Anda, misalnya arn:aws:iam: :111122223333:. role/partner com/rolename. Ini akan membantu membedakan sumber daya yang terkait dengan integrasi Anda dan membuat penemuan, audit, dan analisis lebih mudah bagi pelanggan.

Cross-account persyaratan akses

Pernyataan yang berpotensi mengizinkan akses lintas akun harus menyertakan setidaknya salah satu dari berikut ini:

- Kondisi yang menentukan akun atau organisasi untuk sumber daya (misalnya, aws: men ResourceOrgId cocokkan satu atau lebih nilai yang diharapkan)
- Bidang Sumber daya yang menyertakan akun tertentu (misalnya, arn:aws:sqs: *:111122223333: *)
- Bidang Sumber daya yang menyertakan akun non-wildcard dan nama sumber daya lengkap (misalnya, arn:aws:s3: ::full-bucket-name)

> **Note**
>
> Cross-account akses adalah kemampuan sensitif yang membutuhkan justifikasi bisnis yang jelas. AWS akan meninjau dengan cermat kebutuhan akses lintas akun selama proses orientasi.

Templat kebijakan

Template kebijakan adalah konstruksi IAM baru yang dirancang untuk menentukan izin sementara yang diminta mitra di akun pelanggan. Seperti kebijakan IAM biasa, mereka menentukan izin menggunakan pernyataan dengan elemen Efek, Tindakan, Sumber Daya, dan Kondisi. Perbedaan utamanya adalah template kebijakan menyertakan parameter (seperti @ {bucketName}) yang diganti dengan nilai aktual saat Anda membuat permintaan delegasi.

Cara Kerja Template Kebijakan

Sebagai bagian dari proses orientasi, Anda mendaftarkan template kebijakan Anda dengan AWS. AWS menetapkan setiap template ARN unik yang Anda referensikan saat membuat permintaan delegasi.

Saat membuat permintaan delegasi, Anda menentukan:

- Template kebijakan ARN
- Nilai parameter untuk diganti ke dalam template

AWS menggabungkan template dengan nilai parameter Anda untuk menghasilkan kebijakan IAM standar. Pelanggan meninjau kebijakan render akhir ini saat menyetujui permintaan delegasi Anda, melihat dengan tepat izin apa yang akan diberikan.

> **Note**
>
> Kebijakan render akhir memiliki batas ukuran maksimum 2048 karakter.

Berikut adalah contoh sederhana yang menunjukkan cara kerja substitusi template.

Templat Kebijakan:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

Parameter yang Disediakan dalam Permintaan Delegasi:

```
{
    "Name": "bucketName",
    "Values": ["customer-data-bucket"],
    "Type": "String"
}
```

Kebijakan yang diberikan akhir (apa yang dilihat pelanggan):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::customer-data-bucket/*"
        }
    ]
}
```

Sintaks Template

Template kebijakan menggunakan dua fitur utama untuk memberikan fleksibilitas: substitusi parameter dan pernyataan bersyarat. Substitusi parameter memungkinkan Anda untuk menentukan placeholder di template Anda yang diganti dengan nilai aktual saat membuat permintaan delegasi. Pernyataan bersyarat memungkinkan Anda untuk menyertakan atau mengecualikan seluruh pernyataan kebijakan berdasarkan nilai parameter.

Substitusi dan Jenis Parameter

Gunakan sintaks @ {parameterName} untuk menentukan parameter dalam template kebijakan Anda. Saat membuat permintaan delegasi, Anda harus menentukan jenis untuk setiap parameter.

String

Nilai tunggal yang diganti langsung ke dalam template.

Template:

```
"Resource": "arn:aws:s3:::@{bucketName}/*"
```

Parameter:

```
{
    "Name": "bucketName",
    "Values": ["my-bucket"],
    "Type": "String"
}
```

Hasil yang dirender:

```
"Resource": "arn:aws:s3:::my-bucket/*"
```

StringList

Beberapa nilai yang menghasilkan beberapa entri sumber daya. Ketika StringList parameter digunakan dalam sumber daya ARN, itu diperluas untuk membuat entri sumber daya terpisah untuk setiap nilai.

Template:

```
"Resource": "arn:aws:s3:::@{bucketNames}/*"
```

Parameter:

```
{
    "Name": "bucketNames",
    "Values": ["bucket-1", "bucket-2"],
    "Type": "StringList"
}
```

Hasil yang dirender:

```
"Resource": [
    "arn:aws:s3:::bucket-1/*",
    "arn:aws:s3:::bucket-2/*"
]
```

Cross-Product Perilaku

Ketika beberapa parameter digunakan dalam sumber daya yang sama ARN, StringList parameter membuat produk silang dari semua kombinasi.

Template:

```
"Resource": "arn:aws:s3:::@{bucketNames}/@{prefix}/*"
```

Parameter:

```
[
```

```
    {
        "Name": "bucketNames",
        "Values": ["bucket-1", "bucket-2"],
        "Type": "StringList"
    },
    {
        "Name": "prefix",
        "Values": ["data"],
        "Type": "String"
    }
]
```

Hasil yang dirender:

```
"Resource": [
    "arn:aws:s3:::bucket-1/data/*",
    "arn:aws:s3:::bucket-2/data/*"
]
```

Pernyataan Bersyarat

Gunakan direktif @Enabled untuk memasukkan atau mengecualikan seluruh pernyataan secara kondisional berdasarkan nilai parameter.

Sintaksis:

- @Enabled: “ParameterName” - Sertakan pernyataan ketika nilai parameter adalah “Benar”
- @Enabled: “! ParameterName” - Sertakan pernyataan ketika nilai parameter TIDAK “Benar” (negasi)

Template:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
```

```
            "@Enabled": "ENABLE_S3_WRITE",
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

Parameter (ketika ENABLE_S3_WRITE adalah “Benar”):

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["True"],
        "Type": "String"
    }
]
```

Hasil yang dirender:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::my-bucket/*"
        }
    ]
}
```

Parameter (ketika ENABLE_S3_WRITE adalah “Salah”):

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["False"],
        "Type": "String"
    }
]
```

Hasil yang dirender:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        }
    ]
}
```

Ketika ENABLE_S3_WRITE diatur ke “True”, pernyataan kondisional disertakan. Saat disetel ke “Salah”, pernyataan dikecualikan dari kebijakan yang diberikan.

Contoh Tambahan

Contoh berikut menunjukkan pola umum untuk menggunakan templat kebijakan dalam delegasi sementara. Mereka fokus pada pembuatan peran IAM dengan batas izin untuk akses jangka panjang dan menunjukkan strategi yang berbeda untuk cakupan izin ke sumber daya tertentu. Contoh-contoh ini menggambarkan bagaimana menyeimbangkan fleksibilitas dengan keamanan dengan menggunakan teknik seperti awalan ARN, penandaan sumber daya, dan pembaruan batas izin.

Contoh 1: Memberikan Long-Term Akses ke Sumber Daya Tertentu

Batas izin berikut dikirimkan sebagai "SQSAccessorBoundary" untuk “partner.com”:

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

> **Note**
>
> Ini termasuk kondisi akun yang sama untuk menghindari pemberian akses ke antrian di akun lain dengan kebijakan sumber daya terbuka. Referensi langsung ke ID akun pelanggan tidak dapat disertakan karena batas dibagikan di semua pelanggan dan tidak dapat diberi templat.

Karena ini adalah versi pertama dari kebijakan ini, ARN-nya adalah arn:aws:iam: :partner: -. policy/ permissions boundary/partner com/SQSAccessorBoundary_2025_01_15

Template kebijakan berikut dikirimkan untuk izin akses sementara:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ListQueues"
            ],
            "Resource": "arn:aws:sqs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
```

```
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

Contoh 2: Menggunakan Awalan ARN

Batas izin dapat menentukan awalan ARN sumber daya untuk membatasi akses:

```
"Resource": "arn:aws:sqs:*:@{AccountId}:PartnerPrefix*"
```

Ini membatasi akses hanya ke sumber daya dengan awalan itu, mengurangi ruang lingkup sumber daya yang dapat diakses.

Contoh 3: Menggunakan Tag untuk Kontrol Akses Sumber Daya

Anda dapat menandai sumber daya selama akses delegasi sementara dan mengandalkan tag tersebut untuk kontrol akses jangka panjang.

Batas izin yang memungkinkan akses ke sumber daya yang ditandai:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:ResourceTag/ManagedByPartnerDotCom": "false"
        },
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
```

```
        }
    }
}
```

Template kebijakan untuk menandai antrian baru saat pembuatan:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:CreateQueue",
        "sqs:TagQueue"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:RequestTag/ManagedByPartnerDotCom": "false"
        }
    }
}
```

Template kebijakan untuk menandai antrian yang sudah ada sebelumnya dan membuat peran:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:TagQueue"
            ],
            "Resource": "arn:aws:sqs:*:@{AccountId}:@{QueueName}",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "ManagedByPartnerDotCom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
```

```
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

Pendekatan ini memungkinkan pelanggan untuk secara eksplisit mengkonfirmasi sumber daya spesifik mana yang dapat diakses dalam jangka panjang.

Contoh 4: Memperbarui Batas Izin

Untuk memperbarui batas izin, daftarkan versi baru dengan akhiran tanggal baru dan minta izin untuk menggantinya.

Diperbarui batas izin dengan izin tambahan:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:PurgeQueue",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

Sebagai versi kedua, kebijakan ini memiliki ARN: arn:aws:iam: :partner: -. policy/permissions boundary/partner com/SQSAccessorBoundary_2025_01_20

Template kebijakan untuk memperbarui batas izin pada peran yang ada:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePermissionsBoundary"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20"
                }
            }
        }
    ]
}
```

Pelanggan harus menyetujui permintaan delegasi ini untuk memperbarui batas izin pada peran yang ada.

Membangun integrasi Anda

Memahami Siklus Hidup Permintaan

Sebelum membangun integrasi, penting untuk memahami bagaimana proses permintaan delegasi mulai dari pembuatan hingga penyelesaian.

Negara Permintaan

Permintaan delegasi berlangsung melalui status berikut:

Status	Deskripsi
Tidak ditugaskan	Permintaan dibuat tetapi belum dikaitkan dengan akun pelanggan dan prinsipal IAM. Permintaan mungkin dibuat tanpa menentukan akun target, atau dengan ID akun target tetapi belum diklaim oleh pemilik akun.
Ditugaskan	Permintaan terkait dengan akun pelanggan dan menunggu peninjauan

Status	Deskripsi
Menunggu Persetujuan	Pelanggan telah meneruskan permintaan ke administrator untuk persetujuan
Diterima	Permintaan disetujui oleh pelanggan tetapi token pertukaran belum dirilis
Selesai	Token pertukaran dirilis ke penyedia produk. Periode delegasi (validitas token pertukaran) dimulai ketika permintaan mencapai status Selesai
Ditolak	Permintaan ditolak oleh pelanggan
Kedaluwarsa	Permintaan kedaluwarsa karena tidak aktif atau batas waktu

Transisi Negara

Aliran Normal (Jalur Persetujuan)

- Tidak ditugaskan → Ditugaskan: Pelanggan mengaitkan permintaan dengan akun mereka
- Ditugaskan → Diterima ATAU Ditugaskan → Persetujuan Tertunda: Pelanggan menyetujui permintaan secara langsung ATAU meneruskan ke administrator untuk ditinjau
- Persetujuan Tertunda → Diterima: Administrator menyetujui permintaan
- Diterima → Selesai: Pelanggan merilis token pertukaran

Jalur Penolakan

- Ditugaskan → Ditolak: Pelanggan menolak permintaan
- Persetujuan Tertunda → Ditolak: Administrator menolak permintaan
- Diterima → Ditolak: Pelanggan mencabut persetujuan sebelum melepaskan token

Jalur Kedaluwarsa

Permintaan secara otomatis kedaluwarsa jika tidak ada tindakan yang diambil dalam jangka waktu yang ditentukan:

- Tidak Ditugaskan → Kedaluwarsa (1 hari)
- Ditugaskan → Kedaluwarsa (7 hari)

- Menunggu Persetujuan → Kedaluwarsa (7 hari)
- Diterima → Kedaluwarsa (7 hari)
- Ditolak → Kedaluwarsa (7 hari)
- Selesai → Kedaluwarsa (7 hari)

Negara Terminal

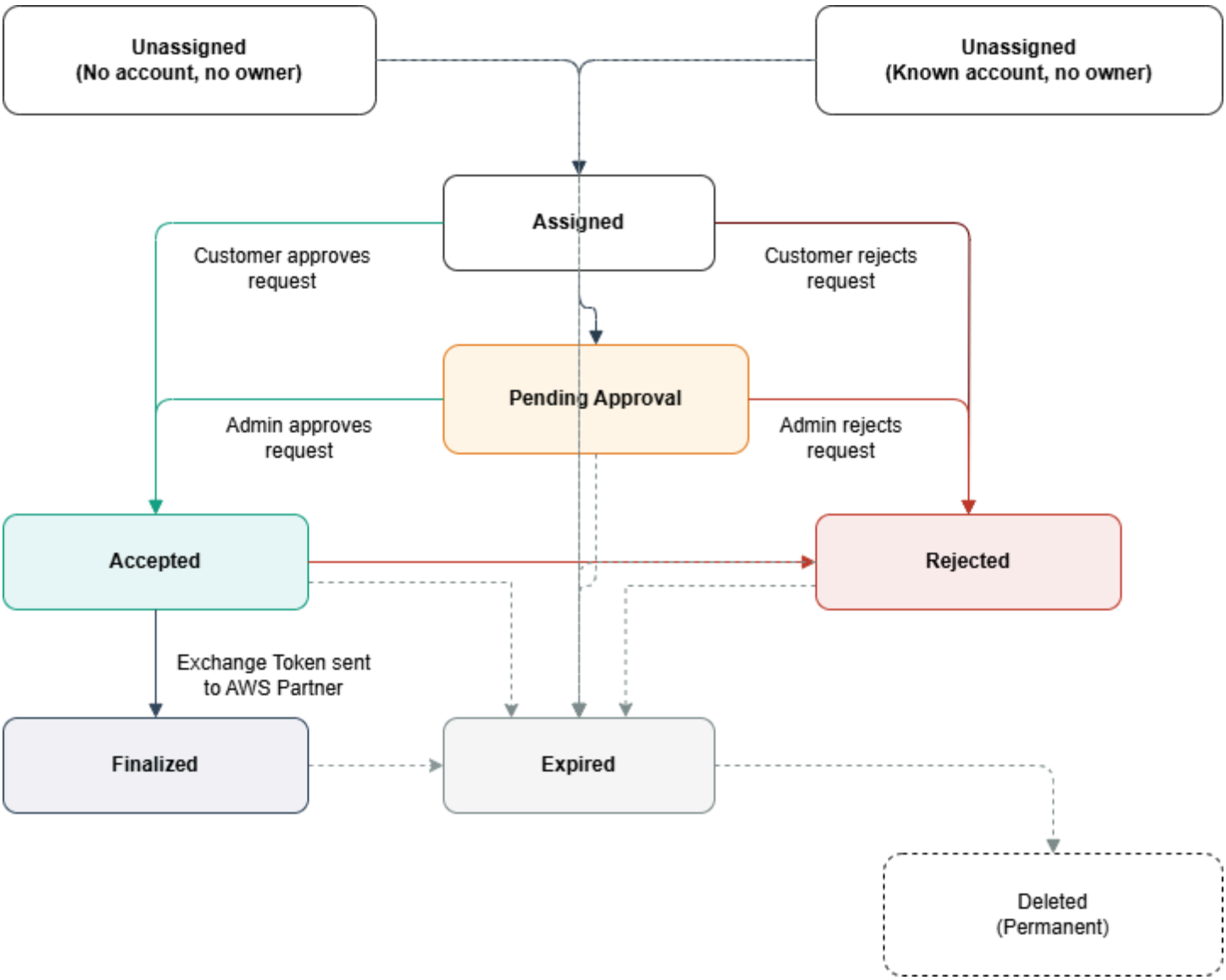
Status berikut adalah terminal (tidak ada transisi lebih lanjut):

- Selesai: Token Exchange dikirim
- Ditolak: Permintaan ditolak
- Kedaluwarsa: Waktu permintaan habis atau periode delegasi berakhir

Permintaan kedaluwarsa akhirnya dihapus dari sistem setelah periode retensi.

Mengelola status permintaan delegasi dalam aplikasi Anda

Sebagai mitra, Anda harus melacak status permintaan delegasi di sistem Anda dan menampilkannya kepada pelanggan Anda. Saat Anda menerima pemberitahuan SNS tentang perubahan status, simpan pembaruan ini di backend Anda dan sertakan di UI yang menghadap pelanggan Anda. Berikan perhatian khusus pada status Persetujuan Tertunde—ketika pelanggan meneruskan permintaan ke administrator untuk ditinjau, AWS mengirimkan pemberitahuan Persetujuan Tertunda kepada Anda. Permintaan dapat tetap dalam status ini hingga 7 hari sambil menunggu tindakan administrator. Selama waktu ini, tunjukkan kepada pelanggan bahwa permintaan mereka sedang menunggu persetujuan administrator dalam aplikasi Anda. Pertimbangkan untuk menyediakan tautan mendalam ke Kon AWS sol tempat pelanggan dapat memeriksa status permintaan atau menindaklanjuti dengan administrator mereka. Menangani mesin status dengan benar di backend Anda dan menampilkan informasi status yang benar kepada pelanggan di setiap tahap penting untuk pengalaman integrasi yang baik.



Mengkonfigurasi Pemberitahuan

IAM menggunakan Amazon Simple Notification Service (SNS) untuk mengkomunikasikan perubahan status permintaan delegasi kepada Anda. Saat membuat permintaan delegasi, Anda harus memberikan topik SNS ARN dari AWS akun terdaftar Anda. IAM akan menerbitkan pesan ke topik ini untuk acara penting, termasuk ketika pelanggan menyetujui atau menolak permintaan dan ketika token pertukaran siap.

> **Note**
>
> Topik SNS tidak boleh ada di Wilayah pilihan. AWS Topik SNS Anda harus berada di Wil AWS ayah yang diaktifkan secara default. Untuk daftar Wilayah pilihan, lihat Mengelola Wil AWS ayah di panduan Manajemen AWS Akun.

Konfigurasi Topik SNS

Untuk menerima pemberitahuan permintaan delegasi, Anda harus mengonfigurasi topik SNS untuk memberikan izin IAM untuk menerbitkan pesan ke topik tersebut. Tambahkan pernyataan kebijakan berikut ke kebijakan topik SNS Anda:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIAMServiceToPublish",
            "Effect": "Allow",
            "Principal": {
                "Service": "iam.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:TOPIC-NAME"
        }
    ]
}
```

> **Important**
>
> Topik SNS harus ada di salah satu AWS akun terdaftar Anda. IAM tidak akan menerima topik SNS dari akun lain. Jika kebijakan topik tidak dikonfigurasi dengan benar, Anda tidak akan menerima pemberitahuan perubahan status atau token pertukaran.

Jenis Pemberitahuan

IAM mengirimkan dua jenis notifikasi:

StateChange Pemberitahuan

Dikirim saat permintaan delegasi beralih ke status baru (Ditugaskan, Menunggu Persetujuan, Diterima, Selesai, Ditolak, Kedaluwarsa).

ExchangeToken Pemberitahuan

Dikirim saat pelanggan melepaskan token delegasi (status Finalized). Pemberitahuan ini menyertakan token pertukaran yang Anda butuhkan untuk mendapatkan kredenSIAL.

Negara Pemberitahuan

Anda akan menerima pemberitahuan untuk status permintaan delegasi berikut:

Status	Jenis Pemberitahuan	Deskripsi
DITUGASKAN	StateChange	Permintaan telah dikaitkan dengan akun pelanggan
MENUNGGU PERSETUJUAN	StateChange	Pelanggan telah meneruskan permintaan ke administrator untuk persetujuan
DITERIMA	StateChange	Pelanggan telah menyetujui permintaan tetapi belum merilis token
DISELESAIKAN	StateChange	Pelanggan telah merilis token pertukaran
DISELESAIKAN	ExchangeToken	Pemberitahuan ini berisi Token Exchange
DITOLAK	StateChange	Pelanggan telah menolak permintaan
KEDALUWARSA	StateChange	Permintaan kedaluwarsa sebelum selesai

Format Pesan Pemberitahuan

IAM menerbitkan pemberitahuan SNS standar. Informasi permintaan delegasi terkandung dalam bidang Pesan sebagai string JSON.

Bidang Umum (Semua Pemberitahuan)

Bidang	Tipe	Deskripsi
Tipe	String	Entah "StateChange" atau "ExchangeToken”
RequestId	String	ID permintaan delegasi IAM
Requestor WorkflowId	String	ID alur kerja yang Anda berikan saat membuat permintaa n
Status	String	Status permintaan saat ini
OwnerAccountId	String	ID AWS akun nasabah
UpdatedAt	String	Stempel waktu saat keadaan berubah (format ISO 8601)

Bidang Tambahan (ExchangeToken Hanya Pemberitahuan)

Bidang	Tipe	Deskripsi
ExchangeToken	String	Token untuk ditukar dengan kredenSIAL menggunakan AWS STS GetDelegatedAccessToken API
ExpiresAt	String	Ketika akses yang didelegasikan kedaluwarsa (format ISO 8601)

Contoh Pemberitahuan

StateChange Pemberitahuan

```
{
  "Type": "Notification",
  "MessageId": "61ee8ad4-6eec-56b5-8f3d-eba57556aa13",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestorWorkflowId\":\"workflow-12345\",\"Type\":\"StateChange\",
\"RequestId\":\"dr-abc123\",\"State\":\"ACCEPTED\",\"OwnerAccountId\":\"111122223333\",
\"UpdatedAt\":\"2025-01-15T10:30:00.123Z\"}",
  "Timestamp": "2025-01-15T10:30:00.456Z",
  "SignatureVersion": "1",
  "Signature": "...",
```

```
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```

ExchangeToken Pemberitahuan

```
{
  "Type": "Notification",
  "MessageId": "e44e5435-c72c-5333-aba3-354406782f5b",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestId\":\"dr-abc123\",\"RequestorWorkflowId\":\"workflow-12345\",
\"State\":\"FINALIZED\",\"OwnerAccountId\":\"111122223333\",\"ExchangeToken\":
\"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...\",\"ExpiresAt\":\"2025-01-15T18:30:00.123Z\",
\"UpdatedAt\":\"2025-01-15T10:30:00.456Z\",\"Type\":\"ExchangeToken\"}",
  "Timestamp": "2025-01-15T10:30:00.789Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```

Tukarkan Token

Token pertukaran atau token perdagangan dikeluarkan oleh IAM ketika pelanggan menerima dan menyelesaikan permintaan delegasi. Penyedia produk menggunakan pertukaran atau token perdagangan ini untuk memanggil AWS STS GetDelegatedAccessToken API untuk mendapatkan AWS kredenSIAL sementara dengan izin yang disetujui pelanggan. Token pertukaran itu sendiri tidak memberikan akses ke AWS sumber daya Anda; itu harus ditukar dengan kredenSIAL aktual melalui AWS STS.

Token pertukaran hanya dapat ditukarkan oleh akun penyedia produk yang membuat permintaan delegasi. Akun yang diminta disematkan dalam token, memastikan bahwa hanya penyedia produk resmi yang dapat memperoleh kredenSIAL untuk mengakses akun pelanggan.

Durasi Akses

Periode delegasi dimulai saat pelanggan melepaskan token pertukaran, bukan saat penyedia produk menebusnya. Setelah pelanggan melepaskan token:

- Penyedia produk menerima token melalui notifikasi SNS
- Mereka dapat segera menukarnya dengan kredenSIAL

- Kredensi kedaluwarsa pada: waktu rilis+durasi yang disetujui
- Penyedia produk dapat menukar token beberapa kali sebelum kedaluwarsa untuk mendapatkan kredenSIAL baru jika diperlukan

Penukaran Ganda

Penyedia produk dapat menukar token beberapa kali selama masa berlaku untuk mendapatkan kredenSIAL baru. Namun, semua kredenSIAL yang diperoleh dari token pertukaran yang sama kedaluwarsa pada saat yang sama, berdasarkan kapan Anda merilis token.

Contoh: Jika Anda menyetujui permintaan delegasi selama 2 jam dan melepaskan token pada pukul 10:00 pagi:

Waktu Pelepasan Token	Waktu Pertukaran Token	Kedaluwarsa Kredensi	Waktu yang Dapat Digunakan
10:00 pagi	10:00 pagi	12:00 sore	2 jam
10:00 pagi	10:20 pagi	12:00 sore	1 jam 40 menit
10:00 pagi	11:40 pagi	12:00 sore	20 menit
10:00 pagi	12:10 sore	Gagal (token kedaluwarsa)	0 menit

Seperti yang ditunjukkan pada tabel, menukar token nanti dalam periode validitas menghasilkan waktu yang kurang dapat digunakan untuk penyedia produk.

Manajer akses akun

Apa itu manajer akses akun?

Manajer akses akun memungkinkan Anda menetapkan Akun AWS akses ke pengguna dan grup di instans organisasi Pusat Identitas IAM Anda. Anda menetapkan akses menggunakan peran IAM di. Akun AWS Anda dapat menggunakan manajer akses akun bersama kumpulan izin Pusat Identitas IAM atau sendiri. Pengelola akses akun dan kumpulan izin Pusat Identitas IAM menggunakan pengguna dan grup yang Anda sinkronkan dari sumber identitas atau dibuat langsung di Pusat Identitas IAM.

Mengapa menggunakan manajer akses akun?

Manajer akses akun memungkinkan Anda menetapkan peran IAM yang ada kepada pengguna dan grup yang telah Anda sinkronkan ke Pusat Identitas IAM dari sumber identitas perusahaan Anda. Menggunakan peran IAM yang ada memberi Anda fleksibilitas ekstra untuk menyesuaikan akses pengguna, termasuk:

- Konfigurasikan tag peran IAM dan gunakan tag sesi untuk kontrol akses berbasis atribut (ABAC).
- Konfigurasikan kebijakan kepercayaan peran IAM untuk membatasi asumsi peran atau menggunakan IdP-asserted klaim sebagai kondisi kebijakan kepercayaan.
- Konfigurasikan jalur peran IAM ke peran grup secara logis dan menyederhanakan administrasi. Untuk informasi selengkapnya, lihat Meng onfigurasi jalur peran IAM ke peran grup secara logis dan menyederhanakan administrasi.

Tabel berikut membandingkan tiga opsi akses akun:

	Peran IAM+manajer akses akun	Kumpulan izin Pusat Identitas IAM	Peran IAM+federasi IAM (asumsi peran langsung)
Terbaik untuk	Peran IAM khusus yang bervariasi di seluruh akun	Akses manusia dasar yang seragam seperti read-only dan admin disediakan sebagai	Cocokan untuk beban kerja

	Peran IAM+manajer akses akun	Kumpulan izin Pusat Identitas IAM	Peran IAM+federasi IAM (asumsi peran langsung)
		peran IAM yang tidak dapat diubah	
Lingkup	Beberapa akun (AWS organisasi)	Beberapa akun (AWS organisasi)	Per akun
Penyediaan peran IAM	Memerlukan infrastruktur sebagai kode (IaC) atau penyediaan manual melalui konsol IAM AWS	Disediakan secara otomatis	Memerlukan infrastruktur sebagai kode (iAC) atau penyediaan manual melalui AWS IAM dan konsol IdP eksternal
Role-to-identity pemetaan berada di	Manajer akses akun	Pusat Identitas IAM	IdP Eksternal
Pengalaman pengguna akhir	Pengguna mengakses AWS akun melalui URL portal akses AWS akun setelah masuk AWS sebagai pengguna Pusat Identitas IAM.	Pengguna mengakses AWS akun melalui portal AWS akses setelah masuk AWS sebagai pengguna Pusat Identitas IAM.	Pengguna mengakses AWS akun melalui aplikasi SAML khusus akun (ikon) di portal IdP mereka.

	Peran IAM+manajer akses akun	Kumpulan izin Pusat Identitas IAM	Peran IAM+federasi IAM (asumsi peran langsung)
AWS CLI pengalaman pengguna	1. Pengguna masuk AWS dan kemudian mengakses akun dengan peran tertentu di browser web. 2. Pengguna menge aws login tik pada baris perintah untuk mengambil kredentif sesi peran IAM dari browser. Selama sesi peran IAM, pengguna dapat terus bekerja dengan AWS CLI tanpa harus melakukan autentikasi ulang pada langkah 1.	Pengguna mengonfig urasi profil termasuk akun yang diinginkan dan pasangan peran dengan mengetikaws configure sso. Jenis pengguna aws sso login untuk memulai sesi baru menggunak an profil. Jika tidak ada sesi Pusat Identitas IAM yang aktif, pengguna melakukan otentikas i di browser dan mengotorisasi akses di sana. Lihat Panduan AWS IAM Identity Center Pengguna untuk detail selengkapnya.	Integrasi kustom menggunakan penyedia kre AWS densia SDK

Kemampuan utama

Portal akses akun

Manajer akses akun menyediakan pengguna dengan portal akses akun tempat mereka dapat melihat dan mengakses yang Akun AWS ditugaskan kepada mereka melalui manajer akses akun.

Akses CLI

Tenaga kerja Anda dapat mengakses Akun AWS melalui AWS CLI menggunakan aws login perintah setelah masuk AWS di browser.

API akses akun

Manajer akses akun memiliki namespace API sendiri,`account-access`, yang terpisah dari API IAM. Anda menggunakan namespace ini di SDK AWS CLI dan.

Memulai dengan manajer akses akun

Berikut ini menguraikan bagaimana Anda bisa memulai dengan manajer akses akun. Ini mengasumsikan peran IAM yang ingin Anda tetapkan sudah ada di masing-masing Akun AWS. Jika peran ini tidak ada, lihat Membuat peran menggunakan kebijakan kepercayaan khusus untuk informasi tentang cara membuat peran baru.

Untuk memulai dengan manajer akses akun

1. Pastikan instans organisasi dari Pusat Identitas IAM diaktifkan dan pengguna dan grup tenaga kerja Anda disediakan di sana. Lihat bagian Memulai dengan Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna untuk informasi selengkapnya. Lihat the section called “Prasyarat dan pertimbangan” nanti dalam topik ini untuk lebih jelasnya.

   > **Note**
   >
   > Jika Anda menggunakan Pusat Identitas IAM yang direplikasi ke beberapa Wil AWS ayah atau dengan kunci yang dikelola pelanggan (CMK), Anda harus menambahkan `kms:Decrypt` izin bagi manajer akses akun untuk menggunakan CMK ini saat menghubungkan ke Pusat Identitas IAM. Untuk informasi selengkapnya, lihat Kebijakan kunci KMS Baseline di Panduan AWS IAM Identity Center Pengguna.

2. Pastikan Anda memiliki izin untuk mengaktifkan manajer akses akun dan mengelola penugasan akun di dalamnya. Untuk informasi selengkapnya, lihat the section called “Identitas dan manajemen akses untuk manajer akses akun”.
3. Aktifkan manajer akses akun melalui konsol atau API.
4. Perbarui firewall dan gateway Anda untuk akses ke portal akses akun.
5. Berikan akses manajer akses akun ke peran IAM dengan memperbarui kebijakan kepercayaan mereka.
6. Mulai menetapkan akses akun ke pengguna dan grup tenaga kerja Anda dari Pusat Identitas IAM.

7. Beri tahu pengguna Anda cara mengakses Akun AWS menggunakan penetapan peran manajer akses akun.

Prasyarat dan pertimbangan

Prasyarat dan pertimbangan berikut berlaku:

- Manajer akses akun tersedia di Wilayah AWS Komersil yang diaktifkan secara default. Opt-in Wilayah Komer GovCloud sional, (AS), China, dan Wilayah non-komersial lainnya tidak didukung.
- Saat pengaktifan, manajer akses akun terhubung ke satu instance organisasi dari Pusat Identitas IAM. Anda harus mengaktifkan AWS organisasi dan instance organisasi dari Pusat Identitas IAM sebelumnya. Instans akun Pusat Identitas IAM tidak didukung.
- Sebaiknya aktifkan pengelola akses akun di Wilayah yang sama AWS tempat Anda mengaktifkan instance Pusat Identitas IAM Anda. Jika Anda belum melakukannya, lihat Per timbangan untuk memilih Wil AWS ayah di Panduan AWS IAM Identity Center Pengguna.
- Instans organisasi Pusat Identitas IAM dan manajer akses akun harus diaktifkan di akun manajemen organisasi Anda.

Aktifkan manajer akses akun

Setelah membuat instance organisasi dari Pusat Identitas IAM, Anda dapat mengaktifkan pengelola akses akun melalui konsol atau API-nya. Anda harus melakukan ini di Wilayah yang sama Akun AWS dengan instance Pusat Identitas IAM Anda.

Sebelum mengaktifkan manajer akses akun, jalankan perintah berikut dari akun manajemen AWS organisasi untuk memberikan akses manajer akses akun ke organisasi Anda:

```
aws organizations enable-aws-service-access --service-principal account-
access.amazonaws.com
```

> **Note**
>
> Sementara konsol pengelola akses akun adalah bagian dari konsol IAM, API-nya berada di namespace terpisah.

Gunakan salah satu metode berikut untuk mengaktifkan pengelola akses akun:

Console

Untuk mengaktifkan pengelola akses akun

1. Masuk ke akun manajemen organisasi Anda.
2. Di panel navigasi AWS Identity and Access Management konsol, pilih Manajer akses akun.
3. Konfirmasikan bahwa Wilayah konsol disetel ke Wilayah utama instans Pusat Identitas IAM Anda.
4. Jika tidak ada instance organisasi dari Pusat Identitas IAM, Anda akan diminta untuk mengaktifkannya sekarang sebelum melanjutkan. Jika tidak, pilih Aktifkan manajer akses akun.

AWS CLI

Pertama temukan ARN dari instance Pusat Identitas IAM dan konfirmasikan kode Wilayah cocok dengan Wilayah utama Pusat Identitas IAM. Lalu, jalankan perintah berikut:

```
aws account-access create-application \
  --region <Region> \
  --identity-source '{
    "identityCenter": {
      "instanceArn": "<IAM_IDENTITY_CENTER_INSTANCE_ARN>"
    }
  }'
```

Memperbarui firewall dan gateway

Portal akses akun memberi pengguna Anda akses masuk tunggal ke yang Akun AWS ditetapkan kepada mereka melalui manajer akses akun.

Jika Anda memfilter akses ke AWS domain atau titik akhir URL tertentu dengan menggunakan solusi pemfilteran konten web seperti firewall generasi berikutnya (NGFW) atau Secure Web Gateway (SWG), Anda harus mengizinkan daftar domain dan titik akhir URL yang terkait dengan portal akses akun.

Daftar berikut menyediakan domain tumpukan ganda dan titik akhir URL untuk ditambahkan ke daftar izin solusi pemfilteran konten web Anda. Manajer akses akun tidak memiliki IPv4-only titik akhir.

Dual-stack izinkan daftar

URL portal akses akun:

- `https://[Tenant-ID].account-access.[Region].app.aws`

API yang dipanggil portal akses akun:

- `[Tenant-ID].account-access.[Region].app.aws/api/*`
- `[Tenant-ID].account-access.[Region].app.aws/auth/*`

Tenant-ID adalah pengenal acak unik yang terkait dengan manajer akses akun. Untuk menemukan nilainya, navigasikan ke tab Peng aturan di konsol pengelola akses akun, dan cari URL Aplikasi di bagian detail instans. Tenant-ID adalah subdomain pertama dalam URL.

Titik akhir layanan manajer akses akun:

- Control-plane: `account-access.[Region].api.aws`
- Waktu pelaksanaan: `account-access-runtime.[Region].api.aws`
- Konsol: `console.aws.amazon.com/account-access`

Jika Anda menggunakan AWS CLI, Anda dapat menemukan ID penyewa dengan menjalankan perintah CLI berikut di manajemen organisasi atau akun administrator yang didelegasikan.

```
aws account-access list-applications
```

Tanggapan berisi informasi tentang manajer akses akun termasuk aplikasi ARN dan ID penyewa.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Pertimbangan untuk mendaftarkan domain dan titik akhir URL

Selain persyaratan allowlist untuk portal akses akun, layanan dan aplikasi lain yang Anda gunakan mungkin memerlukan daftar domain yang diizinkan.

- Anda perlu mengizinkandaftar portal AWS akses dan dependensinya. Untuk informasi selengkapnya, lihat Memperbarui firewall dan gateway untuk mengizinkan akses ke portal AWS akses di Panduan AWS IAM Identity Center Pengguna.
- Untuk mengakses Akun AWS Konsol Manajemen AWS, dan konsol manajer akses akun, Anda harus mengizinkannya daftar domain tambahan. Lihat Pem ec ahan Masalah Konsol Manajemen AWS di Panduan Memulai untuk daftar Konsol Manajemen AWS domain.

Hapus manajer akses akun

Ketika manajer akses akun dihapus, semua data di dalamnya dihapus dan tidak dapat dipulihkan.

Gunakan prosedur berikut untuk menghapus manajer akses akun.

Console

Untuk menghapus instans pengelola akses akun

1. Masuk ke akun manajemen organisasi Anda.
2. Di panel navigasi AWS Identity and Access Management konsol, pilih Manajer akses akun.
3. Konfirmasikan bahwa Wilayah konsol disetel ke Wilayah utama instans manajer akses akun Anda.
4. Pilih tab Pengaturan.
5. Di bagian Hapus manajer akses akun, pilih H apus.
6. Kotak dialog meminta Anda untuk mengonfirmasi. Saat siap, pilih H apus.

AWS CLI

Jalankan perintah berikut untuk menghapus manajer akses akun:

```
aws account-access delete-application \
  --region "<Region>" \
  --application-arn "<account_access_manager_ARN>"
```

Warning

Menghapus manajer akses akun tidak dapat diubah dan memiliki konsekuensi sebagai berikut:

- Semua penugasan peran dihapus secara permanen. Pengguna Anda segera kehilangan kemampuan untuk mengakses Akun AWS dengan peran IAM yang ditetapkan melalui manajer akses akun.
- Manajer akses akun dihapus dari portal AWS akses untuk semua pengguna.
- Sesi aktif peran IAM yang ditetapkan melalui manajer akses akun berlanjut hingga kedaluwarsa, tetapi pengguna tidak dapat memulai sesi baru.
- Jika Anda mengaktifkan kembali pengelola akses akun nanti, Anda harus membuat ulang semua tugas dari awal. Tugas sebelumnya tidak dapat dipulihkan.

Sebelum menghapus, pertimbangkan untuk menonaktifkan akses pengguna di akun individu untuk membatasi akses sementara.

Kelola akses ke Akun AWS

Manajer akses akun terintegrasi dengan Pusat Identitas IAM dan AWS Organizations untuk secara terpusat mengelola penugasan peran IAM yang ada di beberapa peran Akun AWS tanpa mengonfigurasi masing-masing akun Anda secara individual. Dengan pengelola akses akun, Anda menetapkan peran IAM yang ada di pengguna dan grup Pusat Identitas IAM untuk mengontrol akses mereka ke tertentu. Akun AWS Akun AWS

Dengan manajer akses akun, tim Anda dapat membuat peran ??? IAM kustom mereka sendiri Akun AWS, dan Anda sebagai administrator menggunakan manajer akses akun untuk mengelola penugasan pengguna dan grup Pusat Identitas IAM secara terpusat ke peran IAM ini.

Note

Dalam konteks manajer akses akun, istilah pengguna dan grup secara eksklusif merujuk pada pengguna dan grup tenaga kerja di Pusat Identitas IAM.

Administrator yang didelegasikan

Saat Anda mengaktifkan pengelola akses akun AWS , buat itu di akun manajemen AWS Organizations karena manajer akses akun mengelola penetapan peran IAM untuk seluruh AWS Organisasi. Anda dapat memilih untuk mendelegasikan administrasi manajer akses akun ke akun anggota di AWS Organizations.

Jika Anda memilih untuk mendaftarkan akun anggota sebagai administrator yang didelegasikan untuk pengelola akses akun, pengguna di akun ini dapat melakukan sebagian besar tugas administratif pengelola akses akun. Tabel berikut menunjukkan apakah tugas administratif manajer akses akun dapat dilakukan oleh pengguna di akun manajemen organisasi, akun anggota administrasi yang didelegasikan, atau akun anggota lain. Manajemen peran IAM dikecualikan dari daftar tugas administratif karena dilakukan di luar manajer akses akun.

Tugas administratif manajer akses akun	Akun manajemen	Akun administrator yang didelegasikan	Akun Anggota
Kelola penetapan peran IAM*	Ya	Ya	Tidak
Aktifkan manajer akses akun	Ya	Tidak	Tidak
Hapus manajer akses akun	Ya	Tidak	Tidak
Mengaktifkan atau menonaktifkan akses pengguna di akun manajemen	Ya	Tidak	Tidak
Mengaktifkan atau menonaktifkan akses pengguna di akun anggota	Ya	Ya	Tidak
Mendaftarkan atau membatalkan pendaftaran akun anggota sebagai	Ya	Tidak	Tidak

Tugas administratif manajer akses akun	Akun manajemen	Akun administrator yang didelegasikan	Akun Anggota
administrator yang didelegasikan			

*Lihat praktik terbaik untuk administrasi yang didelegasikan mengenai penugasan pengguna dan grup ke akun manajemen.

Daftarkan akun anggota

Untuk mengonfigurasi administrasi yang didelegasikan, Anda harus terlebih dahulu mendaftarkan akun anggota di organisasi Anda sebagai administrator yang didelegasikan. Pengguna di akun anggota yang memiliki izin yang memadai akan memiliki akses administratif ke manajer akses akun.

Manajer akses akun mendukung pendaftaran hanya satu akun anggota sebagai administrator yang didelegasikan pada satu waktu. Anda hanya dapat mendaftarkan akun anggota saat masuk dengan kredenSIAL dari akun manajemen.

Gunakan prosedur berikut untuk mendaftarkan akun anggota di AWS organisasi Anda sebagai administrator yang didelegasikan.

Untuk mendaftarkan akun anggota

1. Masuk ke Konsol Manajemen AWS menggunakan kredenSIAL akun manajemen Anda AWS Organizations. KredenSIAL akun manajemen diperlukan untuk menjalankan RegisterDelegatedAdministrator API.
2. Pilih Wilayah tempat pengelola akses akun diaktifkan dan kemudian buka AWS Identity and Access Management konsol.
3. Pilih Manajer akses akun, lalu pilih tab Pengaturan.
4. Di bagian Administrator yang di delegasikan, pilih Daftar.
5. Pada halaman Daftar administrator yang didelegasikan, pilih Akun AWS yang ingin Anda daftarkan, lalu pilih Daftar.

Menghapus pendaftaran akun anggota

Anda hanya dapat membatalkan pendaftaran akun anggota saat masuk dengan kredenSIAL dari akun manajemen.

Gunakan prosedur berikut untuk membatalkan pendaftaran akun anggota di AWS organisasi Anda yang sebelumnya telah ditetapkan sebagai administrator yang didelegasikan.

> **Important**
>
> Saat membatalkan pendaftaran akun, Anda menghapus kemampuan bagi semua pengguna admin untuk mengelola pengelola akses akun dari akun tersebut. Mereka tidak dapat lagi mengelola Akun AWS akses dari akun ini.
> Operasi ini tidak memengaruhi izin atau penugasan yang dikonfigurasi di manajer akses akun. Pengguna akhir Anda terus mengaksesnya Akun AWS dari portal akses akun.

Untuk membatalkan pendaftaran akun anggota

1. Masuk ke Konsol Manajemen AWS menggunakan kredenSIAL akun manajemen Anda AWS Organizations. KredenSIAL akun manajemen diperlukan untuk menjalankan DeregisterDelegatedAdministrator API.
2. Pilih Wilayah tempat pengelola akses akun diaktifkan, lalu buka AWS Identity and Access Management konsol.
3. Pilih Manajer akses akun, lalu pilih tab Pengaturan.
4. Di bagian Administrator yang didelegasikan, pilih Bat alkan pendaftaran akun.
5. Di kotak dialog Deregister administrator yang didelegasikan, tinjau implikasi keamanan, lalu konfirmasikan bahwa Anda memahami.
6. Pilih Batalkan pendaftaran.

Melihat akun administrator yang didelegasikan terdaftar

Gunakan prosedur berikut untuk menemukan akun anggota mana yang AWS Organizations telah dikonfigurasi sebagai administrator yang didelegasikan untuk manajer akses akun.

Untuk melihat akun anggota terdaftar Anda

1. Masuk ke Konsol Manajemen AWS menggunakan kredenSIAL akun manajemen Anda AWS Organizations.
2. Pilih Wilayah tempat pengelola akses akun diaktifkan dan kemudian buka AWS Identity and Access Management konsol.

3. Pilih Manajer akses akun, lalu pilih tab Pengaturan.
4. Di bagian Administrator yang di delegasikan, Anda dapat melihat detail akun administrator yang didelegasikan saat ini.

Siapkan peran IAM Anda

Sebelum mulai menetapkan peran IAM ke pengguna dan grup Pusat Identitas IAM, Anda perlu:

1. Mengidentifikasi peran IAM yang relevan dalam akun yang diminati, dan
2. Pastikan kebijakan ??? kepercayaan peran IAM berisi Pernyataan yang disediakan di bawah ini.

Jika peran memiliki kebijakan kepercayaan yang ada, tambahkan pernyataan ini ke array Pernyataannya. Jika Anda membuat peran baru, Anda dapat menggunakan seluruh dokumen kebijakan. Pernyataan kebijakan ini memungkinkan pengelola akses akun untuk mengambil peran dan menetapkan konteks atas nama pengguna. `Condition`Blok ini memberikan perlindungan wakil yang membingungkan dengan memastikan bahwa manajer akses akun dapat mengambil peran hanya atas nama akun Anda sendiri dan aplikasi manajer akses akun Anda. Untuk informasi selengkapnya, lihat Memper barui kebijakan kepercayaan peran.

Ganti `<your-account-id>` dengan Akun AWS ID Anda, `<region>` dengan Wilayah utama Pusat Identitas IAM Anda, dan `<application-id>` dengan ID aplikasi manajer akses akun Anda. Anda dapat menemukan aplikasi ARN di halaman Pengaturan di konsol manajer akses akun.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
```

```
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

Opsional - **`sts:TagSession`**. Tambahkan `sts:TagSession` ke `Action` daftar jika Anda ingin atribut pernyataan hulu dari penyedia identitas Anda (misalnya, keanggotaan grup, departemen, pusat biaya) disebarkan sebagai tag utama pada sesi yang dikeluarkan manajer akses akun. Tanpa tindakan ini, asumsi peran masih berhasil; hanya propagasi tag sesi yang ditekan. Tambahkan ketika kebijakan IAM hilir di akun target menggunakan tag utama (`aws:PrincipalTag/<key>`); tinggalkan jika tidak. Pertahankan `Condition` blok yang sama.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

> **Note**
>
> Sebaiknya perbarui alur kerja pembuatan peran IAM untuk menambahkan pernyataan kebijakan kepercayaan ini secara otomatis bila diperlukan.

Tetapkan dan hapus akses

Menetapkan akses pengguna atau grup ke Akun AWS

Anda dapat menetapkan akses pengguna atau grup ke sebuah Akun AWS dengan membuat penugasan antara pengguna atau grup, akun, dan peran IAM yang dapat diambil oleh pengguna atau grup di akun. Dengan demikian, penugasan tunggal adalah triplet yang terdiri dari ID pengguna atau grup, ID akun, dan peran IAM ARN.

Jika Anda ingin melakukan penugasan massal seperti menetapkan peran yang sama ke sejumlah pengguna dan grup, Anda harus membuat setiap tugas secara individual. Anda dapat membuat skrip prosedur ini menggunakan API akses akun, dan AWS SDK/CLI.

Konsol

Konsol menyediakan beberapa cara berbeda untuk menetapkan akses baru tergantung pada apakah Anda ingin mengidentifikasi akun, pengguna, atau grup terlebih dahulu.

> **Note**
>
> Konsol manajer akses akun menyediakan kemampuan pencarian untuk pengguna, grup, dan akun. Untuk mengidentifikasi peran yang ingin Anda tetapkan, Anda harus menemukan namanya (atau ARN) sebelumnya. Konsol manajer akses akun tidak dapat mendeteksi peran IAM di Akun AWS.

Untuk semua prosedur di bagian ini, masuk ke akun manajemen (atau akun admin yang didelegasikan), pilih Wilayah tempat pengelola akses akun diaktifkan, dan arahkan ke Manajer akses akun di konsol IAM.

Accounts tab

Untuk menetapkan akses pengguna atau grup dari tab Akun

1. Pilih tab Akun.
2. Jika Anda sudah mengetahui ID akun, pilih Tet apkan akses baru. Pada halaman Tetapkan akses akun, cari pengguna atau grup yang diinginkan, masukkan ID akun, dan nama peran IAM, lalu pilih Tet apkan akses.
3. Jika Anda perlu menemukan akun terlebih dahulu, Anda dapat mencari yang diinginkan Akun AWS dengan nama, ID, atau alamat emailnya. Atau, Anda dapat meninjau struktur organisasi atau daftar akun untuk menemukan yang relevan Akun AWS. Pilih tombol radio di depan akun dan kemudian Tet apkan akses baru. Pada halaman Tetapkan akses akun, cari pengguna atau grup yang diinginkan, masukkan nama peran IAM, lalu pilih Tet apkan akses.
4. Struktur organisasi atau daftar akun memungkinkan Anda memilih akun untuk melihat detail akun dan penugasan akses akun. Anda dapat memilih Tetapkan akses baru dari sana juga.

Users tab

Untuk menetapkan akses pengguna dari tab Pengguna

1. Pilih tab Pengguna.
2. Anda dapat mencari pengguna berdasarkan nama pengguna dan nama tampilan atau menemukannya di daftar pengguna yang ditampilkan.
3. Pilih pengguna yang ingin Anda tetapkan akses baru.
4. Halaman berikutnya menunjukkan atribut pengguna, penugasan akun, dan keanggotaan grup.
5. Pilih Tetapkan akses baru.
6. Pada halaman Tetapkan akses akun, masukkan ID akun dan nama peran IAM, lalu pilih Tet apkan akses.

Groups tab

Untuk menetapkan akses grup dari tab Grup

1. Di panel navigasi, pilih Manajer akses akun.
2. Pilih tab Grup.

3. Anda dapat mencari grup berdasarkan nama grup atau menemukannya dalam daftar grup yang ditampilkan.
4. Pilih grup yang ingin Anda tetapkan akses baru.
5. Halaman berikutnya menunjukkan atribut grup, penugasan akun, dan anggota pengguna.
6. Pilih Tetapkan akses baru.
7. Pada halaman Tetapkan akses akun, masukkan ID akun dan nama peran IAM, lalu pilih Tet apkan akses.

AWS CLI

Per AWS CLI intah berikut dapat digunakan untuk membuat penugasan akun untuk pengguna:

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "userId": "<Identity_Center_User_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

Per AWS CLI intah berikut dapat digunakan untuk membuat penugasan akun untuk grup:

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "groupId": "<IDC_GROUP_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
```

```
    }
}'
```

Menghapus akses pengguna atau grup ke Akun AWS

Anda dapat menemukan tugas untuk dihapus dengan beberapa cara.

Konsol

Untuk semua prosedur di bagian ini, masuk ke akun manajemen (atau akun admin yang didelegasikan), pilih Wilayah tempat pengelola akses akun diaktifkan, dan arahkan ke Manajer akses akun di konsol IAM.

Accounts tab

Untuk menghapus akses pengguna atau grup dari tab Akun

1. Pilih tab Akun.
2. Temukan akun yang relevan dan pilih namanya.
3. Di bagian Tugas akses akun, pilih pengguna atau grup yang ingin Anda hapus aksesnya, lalu pilih Batalkan penetapan.
4. Dalam modal, pilih Batalkan penet apan untuk mengonfirmasi.

Users tab

Untuk menghapus akses pengguna dari tab Pengguna

1. Pilih tab Pengguna.
2. Pilih pengguna yang ingin Anda hapus aksesnya.
3. Di bagian Tugas akses akun, pilih akun yang ingin Anda hapus aksesnya, lalu pilih Batalkan penet apan.
4. Dalam modal, pilih Batalkan penet apan untuk mengonfirmasi.

AWS CLI

Untuk menghapus akses dengan CLI, pertama-tama Anda harus menemukan penetapan akun (hak) dan kemudian menggunakan ID-nya dalam perintah berikut. Anda dapat melihat nanti dalam topik ini

cara mencari penugasan akun untuk akun, pengguna, grup, atau peran tertentu. CLI mengacu pada penugasan sebagai hak.

```
aws account-access delete-entitlement \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --entitlement-id "<ENTITLEMENT_ID>"
```

Melihat atau mencari penugasan akun

Anda dapat melihat penugasan akun dengan beberapa cara.

Konsol

Untuk semua prosedur di bagian ini, masuk ke akun manajemen (atau akun admin yang didelegasikan), pilih Wilayah tempat pengelola akses akun diaktifkan, dan arahkan ke Manajer akses akun di konsol IAM.

Accounts tab

Melihat akses pengguna atau grup dari tab Akun

1. Pilih tab Akun.
2. Secara default, Anda melihat tampilan Hierarchy. Anda dapat beralih ke tampilan Daftar jika diinginkan. Anda dapat mencari akun berdasarkan nama, ID, email, dan ID OU. Ketika Anda menemukan akun yang diinginkan, pilih namanya.
3. Pada halaman akun, Anda dapat melihat penugasan akun di bagian Pen ugasan akses akun. Anda dapat memfilternya lebih lanjut berdasarkan nama pengguna, ID pengguna, nama grup, ID grup, dan peran IAM ARN.

Users tab

Lihat akses pengguna dari tab Pengguna

1. Pilih tab Pengguna.
2. Cari pengguna berdasarkan nama atau ID pengguna. Ketika Anda menemukan pengguna yang diinginkan, pilih nama pengguna.
3. Pada halaman pengguna, Anda dapat melihat penugasan akun pengguna. Anda dapat memfilternya lebih lanjut berdasarkan ID akun, nama akun, dan peran IAM ARN.

Groups tab

Melihat akses grup dari tab Grup

1. Pilih tab Grup.
2. Cari grup berdasarkan nama atau ID grup. Ketika Anda menemukan grup yang diinginkan, pilih nama grup.
3. Pada halaman grup, Anda dapat melihat penugasan akun grup. Anda dapat memfilternya lebih lanjut berdasarkan ID akun, nama akun, dan peran IAM ARN.

AWS CLI

Perintah berikut menggambarkan cara mengambil tugas akun menggunakan filter yang berbeda. CLI mengacu pada penugasan sebagai hak.

Ambil 100 tugas pertama (hak) untuk akun tertentu

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principalRole": {
        "account": "<ACCOUNT_ID>"
      }
    }' \
    --max-results 100
```

Ambil semua tugas (hak) untuk pengguna tertentu

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "USER",
        "id": "<USER_ID>"
      }
    }'
```

Ambil semua tugas (hak) untuk grup tertentu

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "GROUP",
        "id": "<GROUP_ID>"
      }
    }'
```

Sesi dan kontrol akses

Sesi pemahaman

Pengguna yang mengakses Akun AWS menggunakan tugas di manajer akses akun memulai jenis sesi berikut:

1. Sesi peran IAM - Peran IAM yang diasumsikan pengguna adalah kredensi sementara untuk akses ke AWS sumber daya dengan durasi default 1 jam.
2. Sesi portal akses akun — Pengguna memulai sesi ini saat menggunakan portal akses akun. Ini dikelola sebagai sesi aplikasi oleh IAM Identity Center.
3. AWS sesi portal akses (juga disebut sesi interaktif pengguna) - Sesi ini dikelola oleh Pusat Identitas IAM.
4. Sesi sumber identitas eksternal (IdP atau Microsoft AD) — Sesi ini berlaku ketika instance Pusat Identitas IAM terhubung ke sumber identitas eksternal untuk otentikasi.

Untuk informasi selengkapnya tentang sesi interaktif pengguna dan sesi aplikasi di Pusat Identitas IAM termasuk durasi sesi, lihat Mem ahami sesi otentikasi di Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

> **Important**
>
> Sesi peran IAM yang ditetapkan oleh manajer akses akun beroperasi berdasarkan durasi sesi default 1 jam, terlepas dari durasi sesi maksimum yang ditetapkan dalam peran IAM, status sesi di portal akses akun, dan portal AWS akses.

Cabut akses pengguna

Untuk mencabut akses pengguna yang diasumsikan melalui penetapan peran manajer akses akun, lihat M encabut akses pengguna di Panduan AWS IAM Identity Center Pengguna.

Attribute-based kontrol akses

Anda dapat mengonfigurasi penyedia identitas (IdP) untuk mengirim atribut sebagai tag sesi melalui pernyataan SAML dengan menyetel nama atribut ke`https://aws.amazon.com/SAML/Attributes/AccessControl:TagKey`, di *`TagKey`* mana kunci tag sesi yang ingin Anda isi. Pusat Identitas IAM meneruskan nama atribut dan nilai dari IdP untuk evaluasi kebijakan.

Akses tenaga kerja

Manajer akses akun memungkinkan Anda menetapkan Akun AWS akses ke pengguna dan grup tenaga kerja Anda di Pusat Identitas IAM dengan memetakan mereka ke peran IAM di Anda. Akun AWS Jika Anda baru mengenal Pusat Identitas IAM, lihat Meng atur akses tenaga kerja ke AWS sumber daya di Panduan AWS IAM Identity Center Pengguna untuk informasi tentang sesi otentikasi, manajemen akses pengguna, manajemen kata sandi, dan otentikasi multi-faktor.

Portal pengguna

Untuk mengakses AWS akun yang ditugaskan kepada mereka, pengguna tenaga kerja Anda dapat menavigasi langsung ke portal akses akun menggunakan bookmark, atau meluncurkannya dari dalam portal AWS akses.

Pengguna Anda mungkin berinteraksi dengan dua portal pengguna:

Portal akses akun

> Ini adalah portal (URL) yang memungkinkan pengguna tenaga kerja mengakses yang Akun AWS ditugaskan kepada mereka menggunakan manajer akses akun. Portal ini (URL) dapat ditandai untuk akses langsung dan juga tersedia di portal AWS akses sebagai aplikasi terkel AWS ola.

AWS portal akses

> Ini adalah titik masuk utama AWS untuk pengguna Pusat Identitas IAM. Portal ini menampilkan semua AWS aplikasi yang dapat diakses pengguna, termasuk portal akses akun.

> **Tip**
>
> Kami merekomendasikan bookmark kedua portal untuk navigasi yang lebih cepat. Jika Anda menggunakan penyedia identitas eksternal, Anda juga dapat membuat aplikasi bookmark untuk kedua portal di sana.

Memulai dengan portal akses akun

Menemukan URL portal akses akun

Untuk menemukan URL portal akses akun, pilih tab Pengaturan di konsol pengelola akses akun. Cari URL Aplikasi di bagian Detail Instance. URL mengikuti pola`https://[Tenant-ID].account-access.[Region].app.aws`. Tenant-ID adalah pengenal acak unik yang terkait dengan manajer akses akun.

Jika Anda menggunakan AWS CLI, Anda dapat menemukan ID penyewa dengan menjalankan perintah CLI berikut di manajemen organisasi atau akun administrator yang didelegasikan.

```
aws account-access list-applications
```

Tanggapan berisi informasi tentang manajer akses akun termasuk aplikasi ARN dan ID penyewa. Gunakan ID penyewa dan kode Wilayah manajer akses akun untuk membuat URL aplikasi. Ikuti pola yang diberikan sebelumnya di bagian ini.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Menyiapkan URL portal vanity

Untuk petunjuk penerapan domain vanity kustom, lihat Perutean regional untuk portal AWS akses: Menerapkan domain vanity khusus untuk Pusat Identitas IAM di Blog Keamanan. AWS Anda dapat menyesuaikan instruksi ini dengan pola URL portal akses akun.

Berkomunikasi dengan pengguna Anda

Komunikasikan hal-hal berikut kepada pengguna tenaga kerja Anda:

- URL portal akses akun.
- Portal akses akun tersedia untuk pengguna Anda juga sebagai aplikasi di tab Aplikasi di portal AWS akses.
- Jika Anda mengaktifkan peluncuran portal akses akun dari portal penyedia identitas eksternal atau portal lain di luar AWS, sertakan informasi ini dalam komunikasi.

Akses pengguna ke Akun AWS

Topik ini menjelaskan cara pengguna mengakses yang Akun AWS ditugaskan kepada mereka dengan manajer akses akun.

Mendapatkan URL portal Anda

Anda dapat mengakses yang ditugaskan Akun AWS dengan beberapa cara:

1. Administrator Anda mungkin memberi Anda URL portal akses akun (misalnya, `https://aa-gyxmap389.account-access.us-east-2.app.aws` atau URL vanity).
2. Administrator Anda mungkin memberi Anda URL untuk portal AWS akses utama yang menampilkan semua AWS aplikasi yang ditetapkan untuk Anda, termasuk portal akses akun.
3. Organisasi Anda mungkin telah memungkinkan Anda untuk meluncurkan portal akses akun dari portal penyedia identitas eksternal atau portal lain di luar AWS.

Akses Akun AWS dengan browser web

Portal akses akun menampilkan peran yang Akun AWS Anda miliki akses, dan peran IAM yang dapat Anda ambil di setiap akun.

Untuk mengakses akun di browser

1. Arahkan ke URL portal akses akun dan saat diminta, masuk menggunakan kredensional Anda seperti yang diperintahkan oleh administrator Anda.

   > **Note**
   >
   > Untuk meluncurkan portal akses akun dari portal AWS akses sebagai gantinya, navigasikan ke tab Aplikasi.

2. Halaman browser menampilkan tempat Akun AWS Anda ditugaskan. Anda dapat mencari akun berdasarkan nama akun atau ID.
3. Perluas akun yang menarik untuk melihat semua peran yang dapat Anda ambil di akun.
4. Pilih peran IAM untuk masuk (misalnya, Pengembang). Tab browser baru terbuka di mana Anda masuk ke akun dengan peran yang dipilih.

Akses Akun AWS dengan AWS CLI atau AWS SDK

Anda dapat mengakses AWS layanan secara terprogram dengan menggunakan Perangkat Pengembangan AWS Command Line Interface AWS Perangkat Lunak (SDK) dengan kredentif pengguna dari Pusat Identitas IAM.

Portal akses akun memberi pengguna akses masuk tunggal ke penugasan akun mereka Akun AWS yang dikelola di manajer akses akun. Setelah Anda masuk ke portal akses akun, Anda bisa mendapatkan kredenSIAL sementara. Anda kemudian dapat menggunakan kredenSIAL di AWS CLI AWS SDK atau untuk mengakses sumber daya dalam. Akun AWS

Jika Anda menggunakan AWS CLI untuk mengakses AWS layanan secara terprogram, Anda dapat menggunakan prosedur dalam topik ini untuk memulai akses ke. AWS CLI Untuk informasi tentang AWS CLI, lihat Panduan AWS Command Line Interface Pengguna.

Jika Anda menggunakan AWS SDK untuk mengakses AWS layanan secara terprogram, mengikuti prosedur dalam topik ini juga secara langsung menetapkan otentikasi untuk SDK. AWS Untuk informasi tentang AWS SDK, lihat Panduan Refer AWS ensi SDK dan Alat.

Akun AWS akses dengan Masuk aws (direkomendasikan)

Anda dapat menggunakan aws login perintah untuk mendapatkan akses ke Akun AWS yang ditugaskan di manajer akses akun. Pertama, instal yang terbaru AWS CLI. Kemudian, ikuti prosedur ini:

Untuk mengakses akun dengan AWS CLI

1. Di browser default Anda, masuk ke portal akses akun Anda, dan dari sana ke target Akun AWS menggunakan salah satu peran IAM yang ditetapkan untuk akun tersebut. Ikuti prosedur sebelumnya dalam topik ini.
2. Pada baris perintah, jalankan aws login perintah.
3. Jika Anda belum menetapkan Wilayah default, akan meminta AWS CLI Anda untuk menentukan Wil AWS ayah pilihan Anda (misalnya, us-east-2). AWS CLI Akan mengingat pilihan Anda untuk penggunaan di masa mendatang.
4. Mem AWS CLI buka halaman web di browser default Anda dan meminta Anda untuk melanjutkan sesi aktif yang baru saja Anda mulai atau masuk ke sesi baru.
5. Pada halaman web, pilih ubin dengan sesi yang Anda mulai sebelumnya di browser.
6. Halaman browser mengonfirmasi bahwa kredensi Anda telah berhasil dibagikan dan dapat digunakan hingga sesi Anda berakhir. Anda sekarang dapat menutup tab dengan halaman web dan terus bekerja dengan AWS CLI menggunakan sesi ini.

> **Note**
>
> Perintah IAM Identi Center-specific AWS CLI ty seperti aws sso configure dan aws sso login tidak berfungsi dengan penugasan akun di pengelola akses akun.

Akun AWS akses dengan kunci akses

Anda dapat menggunakan metode penyegaran kredensia manual untuk mendapatkan kredenSIAL sementara untuk peran IAM yang ditetapkan kepada Anda dalam suatu hal tertentu. Akun AWS Untuk melakukannya, Anda menyalin dan menempelkan perintah yang diperlukan untuk kredenSIAL sementara. Dengan metode ini, Anda harus menyegarkan kredenSIAL sementara secara manual.

Anda dapat menjalankan AWS CLI perintah hingga kredenSIAL sementara Anda kedaluwarsa.

Untuk mendapatkan kredenSIAL yang Anda refresh secara manual

1. Masuk ke portal akses akun seperti yang dijelaskan di Ak Akun AWS ses dengan browser web.
2. Temukan tempat Akun AWS Anda ingin mengambil kredentif akses dan perluas untuk menampilkan nama peran IAM (misalnya Pengembang).
3. Pilih Hasilkan kunci akses di sebelah peran yang diinginkan.

> **Note**
>
> Jika Anda tidak melihat Akun AWS tugas apa pun, kemungkinan Anda belum ditugaskan ke akun apa pun. Dalam hal ini, hubungi administrator Anda dan minta mereka untuk menetapkan akses kepada Anda.

4. Di kotak dialog Dapatkan kredenSIAL, pilih PowerShell tab macOS dan Linux , Windows, atau tergantung pada sistem operasi tempat Anda menginstal AWS CLI.
5. Pilih salah satu opsi berikut:

   - Opsi 1: Tetapkan variabel AWS lingkungan

     Pilih opsi ini untuk mengganti semua pengaturan kredensia, termasuk pengaturan apa pun dalam file kredenSIAL dan file konfigurasi. Untuk informasi selengkapnya, lihat Variabel lingkungan untuk mengonfigurasi AWS CLI di Panduan AWS Command Line Interface Pengguna.

     Untuk menggunakan opsi ini, salin perintah ke clipboard Anda, tempel perintah ke jendela AWS CLI terminal Anda, lalu tekan Enter untuk mengatur variabel lingkungan yang diperlukan.

   - Opsi 2: Gunakan nilai individual di klien AWS layanan Anda

     Pilih opsi ini untuk mengakses AWS sumber daya dari klien AWS layanan. Untuk informasi selengkapnya, lihat Alat untuk Membangun AWS.

     Untuk menggunakan opsi ini, salin nilai ke clipboard Anda, tempelkan nilai ke dalam kode Anda, dan tetapkan ke variabel yang sesuai untuk SDK Anda. Untuk informasi selengkapnya, lihat dokumentasi untuk API SDK spesifik Anda.

Membuat tautan pintasan ke AWS Tujuan Konsol Manajemen

Portal akses akun menawarkan opsi untuk membuat tautan dalam yang dapat dibagikan.

Untuk membuat tautan pintasan

1. Saat masuk ke portal akses akun, pilih tombol Buat tautan pintasan.
2. Di kotak dialog:

    a. Pilih Akun AWS menggunakan ID akun atau nama akun. Saat Anda mengetik, menu drop-down menampilkan ID akun yang cocok dan nama yang dapat Anda akses. Anda hanya dapat memilih akun yang dapat Anda akses.

    b. (Opsional) Pilih peran IAM dari daftar drop-down. Ini adalah peran IAM yang ditugaskan kepada Anda untuk akun yang dipilih. Jika Anda mengabaikan memilih peran, pengguna akan diminta untuk memilih salah satu yang ditugaskan kepada mereka untuk akun yang dipilih saat menggunakan tautan pintasan.

> **Note**
>
> Anda tidak dapat memberikan akses baru dengan tautan pintasan. Tautan pintasan hanya berfungsi dengan akun dan peran yang sudah ditetapkan untuk pengguna.

Membangun yang aman AWS Tautan pintasan Konsol Manajemen dengan pengkodean URL

Semua nilai parameter URL, termasuk ID akun, nama peran IAM, dan URL tujuan, harus URL-encoded demikian.

Tautan pintasan memperluas URL portal akses akun dengan jalur berikut:

```
/#/console?account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

URL lengkap di Wil AWS ayah komersial mengikuti pola ini:

```
https://[Tenant-ID].account-access.[Region].app.aws/#/console?
account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

Berikut adalah contoh tautan pintasan yang menandatangani pengguna ke akun 123456789012 dengan nama peran S3FullAccess IAM, dan membawanya ke halaman beranda konsol S3:

```
https://aa-gyxmap389.account-access.us-east-2.app.aws/#/console?
account_id=123456789012&role_name=S3FullAccess&destination=https%3A%2F
%2Fconsole.aws.amazon.com%2Fs3%2Fhome
```

Pertimbangan-pertimbangan

Pertimbangan berikut berlaku untuk penggunaan tautan pintasan:

- Jika Anda berbagi tautan pintasan dengan rekan kerja, mereka harus memiliki tugas akun yang sama untuk dapat menggunakan tautan dengan cara yang sama.
- Tautan pintasan adalah tautan Regional. Jika Wilayah yang direferensikan mengalami gangguan, tautan mungkin tidak berfungsi.
- Tautan pintasan berisi ID akun dan nama peran dalam teks biasa. Pertimbangkan apakah kebijakan organisasi Anda membatasi tempat penyimpanan metadata ini.

Keamanan di manajer akses akun

Bagian ini menyediakan informasi keamanan khusus untuk manajer akses akun. Ini melengkapi informasi diKeamanan, yang mencakup topik keamanan yang berlaku untuk IAM secara keseluruhan.

Identitas dan manajemen akses untuk manajer akses akun

Bagian berikut menjelaskan izin yang Anda perlukan untuk:

- Mengelola manajer akses akun di akun manajemen organisasi Anda
- Mengelola manajer akses akun dari akun administrator yang didelegasikan

Mengelola manajer akses akun dari akun manajemen organisasi Anda

Untuk mengelola setelan pengelola akses akun di akun manajemen AWS organisasi, Anda memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMCreateApplication",
      "Effect": "Allow",
      "Action": "account-access:CreateApplication",
      "Resource": "*"
    },
    {
      "Sid": "AAMApplicationAndEntitlementManagement",
```

```
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteApplication",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:CreateApplication",
        "sso:DeleteApplication",
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances",
        "sso:PutApplicationAccessScope",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
```

```
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentKMSActions",
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "*"
    },
    {
      "Sid": "CreateServiceLinkedRole",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::111122223333:role/aws-service-role/account-
access.amazonaws.com/AWSServiceRoleForAccountAccessManager",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "account-access.amazonaws.com"
        }
      }
    }
```

```
  ]
}
```

Opsional: Untuk membatasi `account-access:CreateApplication` berdasarkan tag, ganti AAMCreateApplication pernyataan dengan:

```
{
  "Sid": "AAMCreateApplicationConstrainedByTag",
  "Effect": "Allow",
  "Action": "account-access:CreateApplication",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Environment": "Production"
    },
    "ForAllValues:StringEquals": {
      "aws:TagKeys": [
        "Environment"
      ]
    }
  }
}
```

Opsional: Untuk mengelola dan mendaftarkan administrator yang didelegasikan untuk manajer akses akun, tambahkan:

```
{
  "Sid": "ManageDelegatedAdministrator",
  "Effect": "Allow",
  "Action": [
    "organizations:DeregisterDelegatedAdministrator",
    "organizations:RegisterDelegatedAdministrator"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "organizations:ServicePrincipal": "account-access.amazonaws.com"
    }
  }
}
```

Untuk menetapkan peran IAM ke pengguna dan grup tenaga kerja Anda dari akun manajemen AWS organisasi tanpa mengelola setelan pengelola akses akun, Anda memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
```

```
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Mengelola manajer akses akun di akun administrasi yang didelegasikan

Untuk mengelola setelan pengelola akses akun dari akun administrasi yang didelegasikan, Anda memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMDelegatedAdminManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
```

```
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
```

```
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Untuk menetapkan peran IAM ke pengguna dan grup tenaga kerja dari akun administrasi yang didelegasikan tanpa mengelola setelan pengelola akses akun, Anda memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
```

```
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Perlindungan data di manajer akses akun

Bagian ini melengkapi informasi yang diberikan dalam Perlindungan data AWS Identity and Access Management dengan detail tambahan khusus untuk manajer akses akun.

Penggunaan data pengguna dan grup IAM Identity Center

Manajer akses akun menggunakan data pengguna dan grup dari Pusat Identitas IAM untuk mengelola penugasan akun. Namun, manajer akses akun hanya menyimpan dan mencatat AWS pengidentifikasi yang dihasilkan (ID pengguna dan ID grup) — tidak menyimpan atau mencatat nama pengguna, alamat email, atau atribut pribadi lainnya.

Pencatatan dan pemantauan di manajer akses akun

Manajer akses akun menggunakan namespace API sendiri (`account-access`) dan mencatat panggilan API melalui CloudTrail.

Untuk informasi selengkap CloudTrail nya, lihat Panduan AWS CloudTrail Pengguna.

Kuota dan batasan di manajer akses akun

Tabel berikut menjelaskan kuota dalam manajer akses akun. Permintaan peningkatan kuota harus berasal dari manajemen atau akun administrator yang didelegasikan. Untuk menambah kuota, lihat Mem inta kenaikan kuota.

Kuota penugasan kelompok

Sumber daya	Kuota bawaan	Dapat ditingkatkan
Jumlah maksimum grup yang ditugaskan ke peran IAM	20	Ya

Batas throttle

Sumber daya	Kuota bawaan	Dapat ditingkatkan
API akses akun	API akses akun memiliki batas throttle kolektif 20 transaksi per detik (TPS). Untuk API baca, Anda dapat membuka kasus dukungan untuk	Ya

Sumber daya	Kuota bawaan	Dapat ditingkatkan
	meminta kenaikan batas. Operasi API tulis memiliki batas 15 panggilan asinkron yang luar biasa. Batas ini tidak dapat dinaikkan.	

Contoh kode untuk IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan IAM dengan kit pengembangan AWS perangkat lunak (SDK).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- Contoh kode untuk IAM menggunakan AWS SDK
  - Contoh dasar untuk penggunaan IAM AWS SDK
    - Halo IAM
    - Pelajari dasar-dasar IAM dengan AWS SDK
    - Tindakan untuk IAM menggunakan AWS SDK
      - Gunakan AddClientIdToOpenIdConnectProviderdengan CLI
      - Gunakan AddRoleToInstanceProfiledengan CLI
      - Gunakan AddUserToGroupdengan CLI
      - Gunakan AttachGroupPolicydengan CLI
      - Gunakan AttachRolePolicydengan AWS SDK atau CLI
      - Gunakan AttachUserPolicydengan AWS SDK atau CLI
      - Gunakan ChangePassworddengan CLI
      - Gunakan CreateAccessKeydengan AWS SDK atau CLI
      - Gunakan CreateAccountAliasdengan AWS SDK atau CLI
      - Gunakan CreateGroupdengan AWS SDK atau CLI
      - Gunakan CreateInstanceProfiledengan AWS SDK atau CLI
      - Gunakan CreateLoginProfiledengan CLI
      - Gunakan CreateOpenIdConnectProviderdengan CLI
      - Gunakan CreatePolicydengan AWS SDK atau CLI
      - Gunakan CreatePolicyVersiondengan AWS SDK atau CLI
      - Gunakan CreateRoledengan AWS SDK atau CLI
      - Gunakan createSamlProvider dengan AWS SDK atau CLI

- Gunakan CreateServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan CreateUserdengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevicedengan CLI
- Gunakan DeactivateMfaDevicedengan CLI
- Gunakan DeleteAccessKeydengan AWS SDK atau CLI
- Gunakan DeleteAccountAliasdengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicydengan CLI
- Gunakan DeleteGroupdengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicydengan CLI
- Gunakan DeleteInstanceProfiledengan AWS SDK atau CLI
- Gunakan DeleteLoginProfiledengan CLI
- Gunakan DeleteOpenIdConnectProviderdengan CLI
- Gunakan DeletePolicydengan AWS SDK atau CLI
- Gunakan DeletePolicyVersiondengan AWS SDK atau CLI
- Gunakan DeleteRoledengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundarydengan CLI
- Gunakan DeleteRolePolicydengan AWS SDK atau CLI
- Gunakan DeleteSamlProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificatedengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificatedengan CLI
- Gunakan DeleteUserdengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundarydengan CLI
- Gunakan DeleteUserPolicydengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevicedengan CLI
- Gunakan DetachGroupPolicydengan CLI
- Gunakan DetachRolePolicydengan AWS SDK atau CLI
- Gunakan DetachUserPolicydengan AWS SDK atau CLI
- Gunakan EnableMfaDevicedengan CLI

- Gunakan GenerateCredentialReportdengan AWS SDK atau CLI

- Gunakan GenerateServiceLastAccessedDetailsdengan CLI
- Gunakan GetAccessKeyLastUseddengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetailsdengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicydengan AWS SDK atau CLI
- Gunakan GetAccountSummarydengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicydengan CLI
- Gunakan GetContextKeysForPrincipalPolicydengan CLI
- Gunakan GetCredentialReportdengan AWS SDK atau CLI
- Gunakan GetGroupdengan CLI
- Gunakan GetGroupPolicydengan CLI
- Gunakan GetInstanceProfiledengan CLI
- Gunakan GetLoginProfiledengan CLI
- Gunakan GetOpenIdConnectProviderdengan CLI
- Gunakan GetPolicydengan AWS SDK atau CLI
- Gunakan GetPolicyVersiondengan AWS SDK atau CLI
- Gunakan GetRoledengan AWS SDK atau CLI
- Gunakan GetRolePolicydengan CLI
- Gunakan GetSamlProviderdengan CLI
- Gunakan GetServerCertificatedengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetailsdengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntitiesdengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatusdengan AWS SDK atau CLI
- Gunakan GetUserdengan AWS SDK atau CLI
- Gunakan GetUserPolicydengan CLI
- Gunakan ListAccessKeysdengan AWS SDK atau CLI
- Gunakan ListAccountAliasesdengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPoliciesdengan CLI
- Gunakan ListAttachedRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListAttachedUserPoliciesdengan CLI
- Gunakan ListEntitiesForPolicydengan CLI

- Gunakan ListGroupPoliciesdengan CLI
- Gunakan ListGroupsdengan AWS SDK atau CLI
- Gunakan ListGroupsForUserdengan CLI
- Gunakan ListInstanceProfilesdengan CLI
- Gunakan ListInstanceProfilesForRoledengan CLI
- Gunakan ListMfaDevicesdengan CLI
- Gunakan ListOpenIdConnectProvidersdengan CLI
- Gunakan ListPoliciesdengan AWS SDK atau CLI
- Gunakan ListPolicyVersionsdengan AWS SDK atau CLI
- Gunakan ListRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListRoleTagsdengan CLI
- Gunakan ListRolesdengan AWS SDK atau CLI
- Gunakan ListSamlProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificatesdengan AWS SDK atau CLI
- Gunakan ListSigningCertificatesdengan CLI
- Gunakan ListUserPoliciesdengan AWS SDK atau CLI
- Gunakan ListUserTagsdengan CLI
- Gunakan ListUsersdengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevicesdengan CLI
- Gunakan PutGroupPolicydengan CLI
- Gunakan PutRolePermissionsBoundarydengan CLI
- Gunakan PutRolePolicydengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundarydengan CLI
- Gunakan PutUserPolicydengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProviderdengan CLI
- Gunakan RemoveRoleFromInstanceProfiledengan CLI
- Gunakan RemoveUserFromGroupdengan CLI
- Gunakan ResyncMfaDevicedengan CLI
- Gunakan SetDefaultPolicyVersiondengan AWS SDK atau CLI
- Gunakan TagRoledengan CLI

- Gunakan TagUserdengan CLI
- Gunakan UntagRoledengan CLI
- Gunakan UntagUserdengan CLI
- Gunakan UpdateAccessKeydengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicydengan CLI
- Gunakan UpdateAssumeRolePolicydengan CLI
- Gunakan UpdateGroupdengan CLI
- Gunakan UpdateLoginProfiledengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprintdengan CLI
- Gunakan UpdateRoledengan CLI
- Gunakan UpdateRoleDescriptiondengan CLI
- Gunakan UpdateSamlProviderdengan CLI
- Gunakan UpdateServerCertificatedengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificatedengan CLI
- Gunakan UpdateUserdengan AWS SDK atau CLI
- Gunakan UploadServerCertificatedengan AWS SDK atau CLI
- Gunakan UploadSigningCertificatedengan CLI

- Skenario untuk menggunakan IAM AWS SDK
  - Membangun dan mengelola layanan tangguh menggunakan AWS SDK
  - Konfigurasikan konektivitas layanan kontainer
  - Buat tugas kontainer untuk tipe peluncuran tanpa server
  - Buat API istirahat dengan integrasi proxy fungsi
  - Buat pengguna IAM read-only dan read-write menggunakan AWS SDK
  - Membuat layanan kontainer untuk instance mesin virtual
  - Membuat ruang kerja pemantauan terkelola
  - Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
  - Membuat fungsi tanpa server pertama Anda
  - Memulai gudang data tanpa server menggunakan CLI
  - Memulai dengan manajemen konfigurasi
  - Memulai dengan perlindungan perangkat internet of things

- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Memulai dengan orkestrasi alur kerja
- Mengelola kunci akses IAM menggunakan AWS SDK
- Mengelola kebijakan IAM menggunakan AWS SDK
- Mengelola peran IAM menggunakan AWS SDK
- Mengelola akun IAM Anda menggunakan AWS SDK
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Memungkinkan AWS Compute Optimizer Fitur otomatisasi untuk menerapkan tindakan yang disarankan
- Kebijakan untuk mengaktifkan Otomasi di seluruh organisasi Anda
- Kebijakan untuk mengaktifkan Otomasi untuk akun Anda
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening
- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening
- Kebijakan untuk memberikan izin peran terkait layanan untuk Otomatisasi Pengoptimalan Komputasi
- Kembalikan versi kebijakan IAM menggunakan AWS SDK
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Mengatur Attribute-Based Access Control untuk DynamoDB menggunakan AWS Command Line Interface v2
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server

- Bekerja dengan DynamoDB Streams dan menggunakan Time-to-Live AWS Command Line Interface v2
- Bekerja dengan API Pembuat Kebijakan IAM menggunakan AWS SDK

- Contoh kode untuk AWS STS memakai AWS SDK
  - Contoh dasar untuk AWS STS memakai AWS SDK
    - Tindakan untuk AWS STS memakai AWS SDK
      - Gunakan AssumeRoledengan AWS SDK atau CLI
      - Gunakan AssumeRoleWithWebIdentitydengan CLI
      - Gunakan DecodeAuthorizationMessagedengan CLI
      - Gunakan GetFederationTokendengan CLI
      - Gunakan GetSessionTokendengan AWS SDK atau CLI
    - Skenario untuk AWS STS memakai AWS SDK
      - Asumsikan peran IAM yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK
      - Konfigurasikan konektivitas layanan kontainer
      - Membangun URL dengan AWS STS untuk pengguna federasi yang menggunakan AWS SDK
      - Buat tugas kontainer untuk tipe peluncuran tanpa server
      - Buat API istirahat dengan integrasi proxy fungsi
      - Membuat layanan kontainer untuk instance mesin virtual
      - Membuat ruang kerja pemantauan terkelola
      - Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
      - Dapatkan token sesi yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK
      - Memulai dengan manajemen konfigurasi
      - Memulai dengan pendaftar kontainer
      - Memulai dengan pesan internet of things
      - Memulai dengan toko fitur pembelajaran mesin
      - Memulai dengan cluster kubernetes terkelola
      - Memulai dengan streaming terkelola
      - Memulai dengan pemberitahuan push

- Memulai dengan mesin pencari dan analitik
- Memulai dengan firewall aplikasi web
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem

Contoh kode untuk IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan IAM dengan kit pengembangan AWS perangkat lunak (SDK).

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

IAM

- Contoh dasar untuk penggunaan IAM AWS SDK
  - Halo IAM
  - Pelajari dasar-dasar IAM dengan AWS SDK
  - Tindakan untuk IAM menggunakan AWS SDK
    - Gunakan AddClientIdToOpenIdConnectProviderdengan CLI
    - Gunakan AddRoleToInstanceProfiledengan CLI
    - Gunakan AddUserToGroupdengan CLI
    - Gunakan AttachGroupPolicydengan CLI
    - Gunakan AttachRolePolicydengan AWS SDK atau CLI

- Gunakan AttachUserPolicydengan AWS SDK atau CLI
- Gunakan ChangePassworddengan CLI
- Gunakan CreateAccessKeydengan AWS SDK atau CLI
- Gunakan CreateAccountAliasdengan AWS SDK atau CLI
- Gunakan CreateGroupdengan AWS SDK atau CLI
- Gunakan CreateInstanceProfiledengan AWS SDK atau CLI
- Gunakan CreateLoginProfiledengan CLI
- Gunakan CreateOpenIdConnectProviderdengan CLI
- Gunakan CreatePolicydengan AWS SDK atau CLI
- Gunakan CreatePolicyVersiondengan AWS SDK atau CLI
- Gunakan CreateRoledengan AWS SDK atau CLI
- Gunakan createSamlProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan CreateUserdengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevicedengan CLI
- Gunakan DeactivateMfaDevicedengan CLI
- Gunakan DeleteAccessKeydengan AWS SDK atau CLI
- Gunakan DeleteAccountAliasdengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicydengan CLI
- Gunakan DeleteGroupdengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicydengan CLI
- Gunakan DeleteInstanceProfiledengan AWS SDK atau CLI
- Gunakan DeleteLoginProfiledengan CLI
- Gunakan DeleteOpenIdConnectProviderdengan CLI
- Gunakan DeletePolicydengan AWS SDK atau CLI
- Gunakan DeletePolicyVersiondengan AWS SDK atau CLI
- Gunakan DeleteRoledengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundarydengan CLI
- Gunakan DeleteRolePolicydengan AWS SDK atau CLI
- Gunakan DeleteSamlProvider dengan AWS SDK atau CLI

- Gunakan DeleteServerCertificatedengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificatedengan CLI
- Gunakan DeleteUserdengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundarydengan CLI
- Gunakan DeleteUserPolicydengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevicedengan CLI
- Gunakan DetachGroupPolicydengan CLI
- Gunakan DetachRolePolicydengan AWS SDK atau CLI
- Gunakan DetachUserPolicydengan AWS SDK atau CLI
- Gunakan EnableMfaDevicedengan CLI
- Gunakan GenerateCredentialReportdengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetailsdengan CLI
- Gunakan GetAccessKeyLastUseddengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetailsdengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicydengan AWS SDK atau CLI
- Gunakan GetAccountSummarydengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicydengan CLI
- Gunakan GetContextKeysForPrincipalPolicydengan CLI
- Gunakan GetCredentialReportdengan AWS SDK atau CLI
- Gunakan GetGroupdengan CLI
- Gunakan GetGroupPolicydengan CLI
- Gunakan GetInstanceProfiledengan CLI
- Gunakan GetLoginProfiledengan CLI
- Gunakan GetOpenIdConnectProviderdengan CLI
- Gunakan GetPolicydengan AWS SDK atau CLI
- Gunakan GetPolicyVersiondengan AWS SDK atau CLI
- Gunakan GetRoledengan AWS SDK atau CLI
- Gunakan GetRolePolicydengan CLI
- Gunakan GetSamlProviderdengan CLI

- Gunakan GetServerCertificatedengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetailsdengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntitiesdengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatusdengan AWS SDK atau CLI
- Gunakan GetUserdengan AWS SDK atau CLI
- Gunakan GetUserPolicydengan CLI
- Gunakan ListAccessKeysdengan AWS SDK atau CLI
- Gunakan ListAccountAliasesdengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPoliciesdengan CLI
- Gunakan ListAttachedRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListAttachedUserPoliciesdengan CLI
- Gunakan ListEntitiesForPolicydengan CLI
- Gunakan ListGroupPoliciesdengan CLI
- Gunakan ListGroupsdengan AWS SDK atau CLI
- Gunakan ListGroupsForUserdengan CLI
- Gunakan ListInstanceProfilesdengan CLI
- Gunakan ListInstanceProfilesForRoledengan CLI
- Gunakan ListMfaDevicesdengan CLI
- Gunakan ListOpenIdConnectProvidersdengan CLI
- Gunakan ListPoliciesdengan AWS SDK atau CLI
- Gunakan ListPolicyVersionsdengan AWS SDK atau CLI
- Gunakan ListRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListRoleTagsdengan CLI
- Gunakan ListRolesdengan AWS SDK atau CLI
- Gunakan ListSamlProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificatesdengan AWS SDK atau CLI
- Gunakan ListSigningCertificatesdengan CLI
- Gunakan ListUserPoliciesdengan AWS SDK atau CLI
- Gunakan ListUserTagsdengan CLI
- Gunakan ListUsersdengan AWS SDK atau CLI

- Gunakan ListVirtualMfaDevicesdengan CLI
- Gunakan PutGroupPolicydengan CLI
- Gunakan PutRolePermissionsBoundarydengan CLI
- Gunakan PutRolePolicydengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundarydengan CLI
- Gunakan PutUserPolicydengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProviderdengan CLI
- Gunakan RemoveRoleFromInstanceProfiledengan CLI
- Gunakan RemoveUserFromGroupdengan CLI
- Gunakan ResyncMfaDevicedengan CLI
- Gunakan SetDefaultPolicyVersiondengan AWS SDK atau CLI
- Gunakan TagRoledengan CLI
- Gunakan TagUserdengan CLI
- Gunakan UntagRoledengan CLI
- Gunakan UntagUserdengan CLI
- Gunakan UpdateAccessKeydengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicydengan CLI
- Gunakan UpdateAssumeRolePolicydengan CLI
- Gunakan UpdateGroupdengan CLI
- Gunakan UpdateLoginProfiledengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprintdengan CLI
- Gunakan UpdateRoledengan CLI
- Gunakan UpdateRoleDescriptiondengan CLI
- Gunakan UpdateSamlProviderdengan CLI
- Gunakan UpdateServerCertificatedengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificatedengan CLI
- Gunakan UpdateUserdengan AWS SDK atau CLI
- Gunakan UploadServerCertificatedengan AWS SDK atau CLI
- Gunakan UploadSigningCertificatedengan CLI

- Skenario untuk menggunakan IAM AWS SDK

- Membangun dan mengelola layanan tangguh menggunakan AWS SDK
- Konfigurasikan konektivitas layanan kontainer
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Buat API istirahat dengan integrasi proxy fungsi
- Buat pengguna IAM read-only dan read-write menggunakan AWS SDK
- Membuat layanan kontainer untuk instance mesin virtual
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai gudang data tanpa server menggunakan CLI
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Memulai dengan orkestrasi alur kerja
- Mengelola kunci akses IAM menggunakan AWS SDK
- Mengelola kebijakan IAM menggunakan AWS SDK
- Mengelola peran IAM menggunakan AWS SDK
- Mengelola akun IAM Anda menggunakan AWS SDK
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Memungkinkan AWS Compute Optimizer Fitur otomatisasi untuk menerapkan tindakan yang disarankan
- Kebijakan untuk mengaktifkan Otomasi di seluruh organisasi Anda
- Kebijakan untuk mengaktifkan Otomasi untuk akun Anda
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening

- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening
- Kebijakan untuk memberikan izin peran terkait layanan untuk Otomatisasi Pengoptimalan Komputasi
- Kembalikan versi kebijakan IAM menggunakan AWS SDK
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Mengatur Attribute-Based Access Control untuk DynamoDB menggunakan AWS Command Line Interface v2
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server
- Bekerja dengan DynamoDB Streams dan menggunakan Time-to-Live AWS Command Line Interface v2
- Bekerja dengan API Pembuat Kebijakan IAM menggunakan AWS SDK

Contoh dasar untuk penggunaan IAM AWS SDK

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar AWS Identity and Access Management dengan AWS SDK.

Contoh

- Halo IAM
- Pelajari dasar-dasar IAM dengan AWS SDK
- Tindakan untuk IAM menggunakan AWS SDK
  - Gunakan AddClientIdToOpenIdConnectProviderdengan CLI
  - Gunakan AddRoleToInstanceProfiledengan CLI
  - Gunakan AddUserToGroupdengan CLI
  - Gunakan AttachGroupPolicydengan CLI
  - Gunakan AttachRolePolicydengan AWS SDK atau CLI
  - Gunakan AttachUserPolicydengan AWS SDK atau CLI
  - Gunakan ChangePassworddengan CLI

- Gunakan CreateAccessKeydengan AWS SDK atau CLI
- Gunakan CreateAccountAliasdengan AWS SDK atau CLI
- Gunakan CreateGroupdengan AWS SDK atau CLI
- Gunakan CreateInstanceProfiledengan AWS SDK atau CLI
- Gunakan CreateLoginProfiledengan CLI
- Gunakan CreateOpenIdConnectProviderdengan CLI
- Gunakan CreatePolicydengan AWS SDK atau CLI
- Gunakan CreatePolicyVersiondengan AWS SDK atau CLI
- Gunakan CreateRoledengan AWS SDK atau CLI
- Gunakan createSamlProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan CreateUserdengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevicedengan CLI
- Gunakan DeactivateMfaDevicedengan CLI
- Gunakan DeleteAccessKeydengan AWS SDK atau CLI
- Gunakan DeleteAccountAliasdengan AWS SDK atau CLI
- Gunakan DeleteAccountPasswordPolicydengan CLI
- Gunakan DeleteGroupdengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicydengan CLI
- Gunakan DeleteInstanceProfiledengan AWS SDK atau CLI
- Gunakan DeleteLoginProfiledengan CLI
- Gunakan DeleteOpenIdConnectProviderdengan CLI
- Gunakan DeletePolicydengan AWS SDK atau CLI
- Gunakan DeletePolicyVersiondengan AWS SDK atau CLI
- Gunakan DeleteRoledengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundarydengan CLI
- Gunakan DeleteRolePolicydengan AWS SDK atau CLI
- Gunakan DeleteSamlProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificatedengan AWS SDK atau CLI

- Gunakan DeleteServiceLinkedRoledengan AWS SDK atau CLI

- Gunakan DeleteSigningCertificatedengan CLI
- Gunakan DeleteUserdengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundarydengan CLI
- Gunakan DeleteUserPolicydengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevicedengan CLI
- Gunakan DetachGroupPolicydengan CLI
- Gunakan DetachRolePolicydengan AWS SDK atau CLI
- Gunakan DetachUserPolicydengan AWS SDK atau CLI
- Gunakan EnableMfaDevicedengan CLI
- Gunakan GenerateCredentialReportdengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetailsdengan CLI
- Gunakan GetAccessKeyLastUseddengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetailsdengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicydengan AWS SDK atau CLI
- Gunakan GetAccountSummarydengan AWS SDK atau CLI
- Gunakan GetContextKeysForCustomPolicydengan CLI
- Gunakan GetContextKeysForPrincipalPolicydengan CLI
- Gunakan GetCredentialReportdengan AWS SDK atau CLI
- Gunakan GetGroupdengan CLI
- Gunakan GetGroupPolicydengan CLI
- Gunakan GetInstanceProfiledengan CLI
- Gunakan GetLoginProfiledengan CLI
- Gunakan GetOpenIdConnectProviderdengan CLI
- Gunakan GetPolicydengan AWS SDK atau CLI
- Gunakan GetPolicyVersiondengan AWS SDK atau CLI
- Gunakan GetRoledengan AWS SDK atau CLI
- Gunakan GetRolePolicydengan CLI
- Gunakan GetSamlProviderdengan CLI
- Gunakan GetServerCertificatedengan AWS SDK atau CLI

- Gunakan GetServiceLastAccessedDetailsdengan CLI

- Gunakan GetServiceLastAccessedDetailsWithEntitiesdengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatusdengan AWS SDK atau CLI
- Gunakan GetUserdengan AWS SDK atau CLI
- Gunakan GetUserPolicydengan CLI
- Gunakan ListAccessKeysdengan AWS SDK atau CLI
- Gunakan ListAccountAliasesdengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPoliciesdengan CLI
- Gunakan ListAttachedRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListAttachedUserPoliciesdengan CLI
- Gunakan ListEntitiesForPolicydengan CLI
- Gunakan ListGroupPoliciesdengan CLI
- Gunakan ListGroupsdengan AWS SDK atau CLI
- Gunakan ListGroupsForUserdengan CLI
- Gunakan ListInstanceProfilesdengan CLI
- Gunakan ListInstanceProfilesForRoledengan CLI
- Gunakan ListMfaDevicesdengan CLI
- Gunakan ListOpenIdConnectProvidersdengan CLI
- Gunakan ListPoliciesdengan AWS SDK atau CLI
- Gunakan ListPolicyVersionsdengan AWS SDK atau CLI
- Gunakan ListRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListRoleTagsdengan CLI
- Gunakan ListRolesdengan AWS SDK atau CLI
- Gunakan ListSamlProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificatesdengan AWS SDK atau CLI
- Gunakan ListSigningCertificatesdengan CLI
- Gunakan ListUserPoliciesdengan AWS SDK atau CLI
- Gunakan ListUserTagsdengan CLI
- Gunakan ListUsersdengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevicesdengan CLI

- Gunakan PutGroupPolicydengan CLI

- Gunakan PutRolePermissionsBoundarydengan CLI
- Gunakan PutRolePolicydengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundarydengan CLI
- Gunakan PutUserPolicydengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProviderdengan CLI
- Gunakan RemoveRoleFromInstanceProfiledengan CLI
- Gunakan RemoveUserFromGroupdengan CLI
- Gunakan ResyncMfaDevicedengan CLI
- Gunakan SetDefaultPolicyVersiondengan AWS SDK atau CLI
- Gunakan TagRoledengan CLI
- Gunakan TagUserdengan CLI
- Gunakan UntagRoledengan CLI
- Gunakan UntagUserdengan CLI
- Gunakan UpdateAccessKeydengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicydengan CLI
- Gunakan UpdateAssumeRolePolicydengan CLI
- Gunakan UpdateGroupdengan CLI
- Gunakan UpdateLoginProfiledengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprintdengan CLI
- Gunakan UpdateRoledengan CLI
- Gunakan UpdateRoleDescriptiondengan CLI
- Gunakan UpdateSamlProviderdengan CLI
- Gunakan UpdateServerCertificatedengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificatedengan CLI
- Gunakan UpdateUserdengan AWS SDK atau CLI
- Gunakan UploadServerCertificatedengan AWS SDK atau CLI
- Gunakan UploadSigningCertificatedengan CLI

Halo IAM

Contoh kode berikut menunjukkan bagaimana memulai menggunakan IAM.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Kode untuk file CMakeLists.txt CMake.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk C++ API.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
```

```
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
```

```
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
    main()
```

- Untuk detail API, lihat ListPoliciesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dari src/bin /hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), Box<SdkError<ListPoliciesError>>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

Dari src/iam -service-lib.rs.

```
pub async fn list_policies(
```

```
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Untuk detail API, lihat ListPoliciesreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Pelajari dasar-dasar IAM dengan AWS SDK

Contoh kode berikut menunjukkan cara membuat pengguna dan mengambil peran.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat pengguna tanpa izin.
- Buat peran yang memberikan izin untuk mencantumkan bucket Amazon S3 untuk akun tersebut.
- Tambahkan kebijakan agar pengguna dapat mengambil peran tersebut.
- Asumsikan peran dan daftar bucket S3 menggunakan kredensyal sementara, lalu bersihkan sumber daya.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
```

```
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
```

```
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };
```

```
        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
```

```
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
```

```
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
 userName)
    {
```

```
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
```

```
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
```

```
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
```

```
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
```

```
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
```

```
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
```

```
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
```

```
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;
```

```
        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
```

```
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
```

```
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");
```

```
        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
```

```
    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
```

```
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;
```

```
public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
 s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
 to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
 role.");
```

```
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
```

```
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk .NET .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
```

```
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
```

```
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
```

```
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
```

```
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Fungsi IAM yang digunakan dalam skenario ini.

```
###############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
```

```
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) key pair."
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
```

```
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
```

```
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi

iecho "delete-access-key response:$response"
iecho
```

```
  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;
```

```
    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }
```

```
    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
```

```
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
```

```
                        listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
```

```
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
```

```
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }
```

```
    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk C++ .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

Jalankan skenario interaktif di penggugah/prompt perintah.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
```

```
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
```

```
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
```

```
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
```

```
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
```

```
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
```

```
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
```

```
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
  if err != nil {
```

```
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
```

```
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Tentukan struct yang membungkus tindakan akun.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
```

```
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Tentukan struct yang membungkus tindakan kebijakan.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
```

```
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
```

```
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
```

```
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan peran.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
```

```
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

```
// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

```
// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Tentukan struct yang membungkus tindakan pengguna.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
```

```
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

```
// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
```

```
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Go .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan pengguna IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
```

```
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
```

```
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
```

```
              Principal: {
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
```

```
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);
```

```
  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }
```

```
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk JavaScript .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan pengguna IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>
```

```
    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}
```

```
suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
```

```
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
```

```
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }
```

```
    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")
```

```
    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkap dan pelajari cara menyiapkan dan menjalankan di Repositori Contoh Kode AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
```

```
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
```

```
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk PHP .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
```

```
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
```

```
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
```

```
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
```

```
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
```

```
            raise

        try:
            for user_pol in user.policies.all():
                user_pol.delete()
                print("Deleted inline user policy.")
            for key in user.access_keys.all():
                key.delete()
                print("Deleted user's access key.")
            user.delete()
            print(f"Deleted {user.name}.")
        except ClientError as error:
            print(
                "Couldn't delete user policy or delete user. Here's why: "
                f"{error.response['Error']['Message']}"
            )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensyal sementara untuk membuat daftar bucket untuk akun.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
```

```
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
```

```
 # @param role_name [String] The name to give the role.
 # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
 # @return [Aws::IAM::Role] The newly created role.
 def create_role(role_name, user)
   trust_policy = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Principal: { 'AWS': user.arn },
       Action: 'sts:AssumeRole'
     }]
   }.to_json
   role = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: trust_policy
   ).role
   @logger.info("Created role #{role.role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't create a role for the demo. Here's why: ")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 else
   role
 end

 # Creates a policy that grants permission to list S3 buckets in the account,
and
 # then attaches the policy to a role.
 #
 # @param policy_name [String] The name to give the policy.
 # @param role [Aws::IAM::Role] The role that the policy is attached to.
 # @return [Aws::IAM::Policy] The newly created policy.
 def create_and_attach_role_policy(policy_name, role)
   policy_document = {
     Version: '2012-10-17',
     Statement: [{
       Effect: 'Allow',
       Action: 's3:ListAllMyBuckets',
       Resource: 'arn:aws:s3:::*'
     }]
   }.to_json
   policy = @iam_client.create_policy(
     policy_name: policy_name,
```

```
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 'sts:AssumeRole',
        Resource: role.arn
      }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end
```

```
 # Creates an Amazon S3 resource with specified credentials. This is separated
into a
 # factory function so that it can be mocked for unit testing.
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == 'AccessDenied'
     puts('Attempt to list buckets with no permissions: AccessDenied.')
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end
```

```
 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Deletes a user. If the user has inline policies or access keys, they are
deleted
 # before the user is deleted.
 #
```

```
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
```

```
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Ruby .
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
```

```
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;
```

```
    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
```

```
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
```

```
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
```

```
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Tindakan untuk IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara melakukan tindakan IAM individual dengan AWS SDK. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Kutipan ini memanggil API IAM dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Anda dapat melihat tindakan dalam konteks diSkenario untuk menggunakan IAM AWS SDK.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi AWS Identity and Access Management API.

Contoh

- Gunakan AddClientIdToOpenIdConnectProviderdengan CLI
- Gunakan AddRoleToInstanceProfiledengan CLI
- Gunakan AddUserToGroupdengan CLI
- Gunakan AttachGroupPolicydengan CLI
- Gunakan AttachRolePolicydengan AWS SDK atau CLI
- Gunakan AttachUserPolicydengan AWS SDK atau CLI
- Gunakan ChangePassworddengan CLI
- Gunakan CreateAccessKeydengan AWS SDK atau CLI
- Gunakan CreateAccountAliasdengan AWS SDK atau CLI
- Gunakan CreateGroupdengan AWS SDK atau CLI
- Gunakan CreateInstanceProfiledengan AWS SDK atau CLI
- Gunakan CreateLoginProfiledengan CLI
- Gunakan CreateOpenIdConnectProviderdengan CLI
- Gunakan CreatePolicydengan AWS SDK atau CLI
- Gunakan CreatePolicyVersiondengan AWS SDK atau CLI
- Gunakan CreateRoledengan AWS SDK atau CLI
- Gunakan createSamlProvider dengan AWS SDK atau CLI
- Gunakan CreateServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan CreateUserdengan AWS SDK atau CLI
- Gunakan CreateVirtualMfaDevicedengan CLI
- Gunakan DeactivateMfaDevicedengan CLI
- Gunakan DeleteAccessKeydengan AWS SDK atau CLI
- Gunakan DeleteAccountAliasdengan AWS SDK atau CLI

- Gunakan DeleteAccountPasswordPolicydengan CLI
- Gunakan DeleteGroupdengan AWS SDK atau CLI
- Gunakan DeleteGroupPolicydengan CLI
- Gunakan DeleteInstanceProfiledengan AWS SDK atau CLI
- Gunakan DeleteLoginProfiledengan CLI
- Gunakan DeleteOpenIdConnectProviderdengan CLI
- Gunakan DeletePolicydengan AWS SDK atau CLI
- Gunakan DeletePolicyVersiondengan AWS SDK atau CLI
- Gunakan DeleteRoledengan AWS SDK atau CLI
- Gunakan DeleteRolePermissionsBoundarydengan CLI
- Gunakan DeleteRolePolicydengan AWS SDK atau CLI
- Gunakan DeleteSamlProvider dengan AWS SDK atau CLI
- Gunakan DeleteServerCertificatedengan AWS SDK atau CLI
- Gunakan DeleteServiceLinkedRoledengan AWS SDK atau CLI
- Gunakan DeleteSigningCertificatedengan CLI
- Gunakan DeleteUserdengan AWS SDK atau CLI
- Gunakan DeleteUserPermissionsBoundarydengan CLI
- Gunakan DeleteUserPolicydengan AWS SDK atau CLI
- Gunakan DeleteVirtualMfaDevicedengan CLI
- Gunakan DetachGroupPolicydengan CLI
- Gunakan DetachRolePolicydengan AWS SDK atau CLI
- Gunakan DetachUserPolicydengan AWS SDK atau CLI
- Gunakan EnableMfaDevicedengan CLI
- Gunakan GenerateCredentialReportdengan AWS SDK atau CLI
- Gunakan GenerateServiceLastAccessedDetailsdengan CLI
- Gunakan GetAccessKeyLastUseddengan AWS SDK atau CLI
- Gunakan GetAccountAuthorizationDetailsdengan AWS SDK atau CLI
- Gunakan GetAccountPasswordPolicydengan AWS SDK atau CLI
- Gunakan GetAccountSummarydengan AWS SDK atau CLI

- Gunakan GetContextKeysForCustomPolicydengan CLI
- Gunakan GetContextKeysForPrincipalPolicydengan CLI
- Gunakan GetCredentialReportdengan AWS SDK atau CLI
- Gunakan GetGroupdengan CLI
- Gunakan GetGroupPolicydengan CLI
- Gunakan GetInstanceProfiledengan CLI
- Gunakan GetLoginProfiledengan CLI
- Gunakan GetOpenIdConnectProviderdengan CLI
- Gunakan GetPolicydengan AWS SDK atau CLI
- Gunakan GetPolicyVersiondengan AWS SDK atau CLI
- Gunakan GetRoledengan AWS SDK atau CLI
- Gunakan GetRolePolicydengan CLI
- Gunakan GetSamlProviderdengan CLI
- Gunakan GetServerCertificatedengan AWS SDK atau CLI
- Gunakan GetServiceLastAccessedDetailsdengan CLI
- Gunakan GetServiceLastAccessedDetailsWithEntitiesdengan CLI
- Gunakan GetServiceLinkedRoleDeletionStatusdengan AWS SDK atau CLI
- Gunakan GetUserdengan AWS SDK atau CLI
- Gunakan GetUserPolicydengan CLI
- Gunakan ListAccessKeysdengan AWS SDK atau CLI
- Gunakan ListAccountAliasesdengan AWS SDK atau CLI
- Gunakan ListAttachedGroupPoliciesdengan CLI
- Gunakan ListAttachedRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListAttachedUserPoliciesdengan CLI
- Gunakan ListEntitiesForPolicydengan CLI
- Gunakan ListGroupPoliciesdengan CLI
- Gunakan ListGroupsdengan AWS SDK atau CLI
- Gunakan ListGroupsForUserdengan CLI
- Gunakan ListInstanceProfilesdengan CLI

- Gunakan ListInstanceProfilesForRoledengan CLI
- Gunakan ListMfaDevicesdengan CLI
- Gunakan ListOpenIdConnectProvidersdengan CLI
- Gunakan ListPoliciesdengan AWS SDK atau CLI
- Gunakan ListPolicyVersionsdengan AWS SDK atau CLI
- Gunakan ListRolePoliciesdengan AWS SDK atau CLI
- Gunakan ListRoleTagsdengan CLI
- Gunakan ListRolesdengan AWS SDK atau CLI
- Gunakan ListSamlProviders dengan AWS SDK atau CLI
- Gunakan ListServerCertificatesdengan AWS SDK atau CLI
- Gunakan ListSigningCertificatesdengan CLI
- Gunakan ListUserPoliciesdengan AWS SDK atau CLI
- Gunakan ListUserTagsdengan CLI
- Gunakan ListUsersdengan AWS SDK atau CLI
- Gunakan ListVirtualMfaDevicesdengan CLI
- Gunakan PutGroupPolicydengan CLI
- Gunakan PutRolePermissionsBoundarydengan CLI
- Gunakan PutRolePolicydengan AWS SDK atau CLI
- Gunakan PutUserPermissionsBoundarydengan CLI
- Gunakan PutUserPolicydengan AWS SDK atau CLI
- Gunakan RemoveClientIdFromOpenIdConnectProviderdengan CLI
- Gunakan RemoveRoleFromInstanceProfiledengan CLI
- Gunakan RemoveUserFromGroupdengan CLI
- Gunakan ResyncMfaDevicedengan CLI
- Gunakan SetDefaultPolicyVersiondengan AWS SDK atau CLI
- Gunakan TagRoledengan CLI
- Gunakan TagUserdengan CLI
- Gunakan UntagRoledengan CLI
- Gunakan UntagUserdengan CLI

- Gunakan UpdateAccessKeydengan AWS SDK atau CLI
- Gunakan UpdateAccountPasswordPolicydengan CLI
- Gunakan UpdateAssumeRolePolicydengan CLI
- Gunakan UpdateGroupdengan CLI
- Gunakan UpdateLoginProfiledengan CLI
- Gunakan UpdateOpenIdConnectProviderThumbprintdengan CLI
- Gunakan UpdateRoledengan CLI
- Gunakan UpdateRoleDescriptiondengan CLI
- Gunakan UpdateSamlProviderdengan CLI
- Gunakan UpdateServerCertificatedengan AWS SDK atau CLI
- Gunakan UpdateSigningCertificatedengan CLI
- Gunakan UpdateUserdengan AWS SDK atau CLI
- Gunakan UploadServerCertificatedengan AWS SDK atau CLI
- Gunakan UploadSigningCertificatedengan CLI

Gunakan **`AddClientIdToOpenIdConnectProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menambahkan ID klien (audiens) ke penyedia Open-ID Connect (OIDC)

`add-client-id-to-open-id-connect-provider`Perintah berikut menambahkan ID klien `my-application-ID` ke penyedia OIDC bernama. `server.example.com`

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Perintah ini tidak menghasilkan output.

Untuk membuat penyedia OIDC, gunakan perintah. `create-open-id-connect-provider`

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat AddClientIdToOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan ID klien (atau audiens) **my-application-ID** ke penyedia OIDC yang ada bernama. **server.example.com**

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Untuk detail API, lihat AddClientIdToOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini menambahkan ID klien (atau audiens) **my-application-ID** ke penyedia OIDC yang ada bernama. **server.example.com**

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Untuk detail API, lihat AddClientIdToOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **AddRoleToInstanceProfile**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AddRoleToInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membuat layanan kontainer untuk instance mesin virtual
- Memulai dengan streaming terkelola
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

CLI

AWS CLI

Untuk menambahkan peran ke profil instance

`add-role-to-instance-profile`Perintah berikut menambahkan peran bernama `S3Access` ke profil instance bernama`Webserver`.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Perintah ini tidak menghasilkan output.

Untuk membuat profil instance, gunakan `create-instance-profile` perintah.

Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2 dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat AddRoleToInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan peran bernama **`S3Access`** ke profil instance yang ada bernama**`webserver`**. Untuk membuat profil instance, gunakan **`New-IAMInstanceProfile`** perintah. Setelah Anda membuat profil instans dan mengaitkannya dengan peran menggunakan perintah ini, Anda dapat melampirkannya ke instans EC2. Untuk melakukan itu, gunakan **`New-EC2Instance`** cmdlet dengan **`InstanceProfile-Name`** parameter **`InstanceProfile_Arn`** atau untuk meluncurkan instance baru.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Untuk detail API, lihat AddRoleToInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini menambahkan peran bernama **`S3Access`** ke profil instance yang ada bernama**`webserver`**. Untuk membuat profil instance, gunakan **`New-IAMInstanceProfile`** perintah. Setelah Anda membuat profil instans dan mengaitkannya dengan peran menggunakan perintah ini, Anda dapat melampirkannya ke instans EC2. Untuk melakukan itu, gunakan **`New-EC2Instance`** cmdlet dengan **`InstanceProfile-Name`** parameter **`InstanceProfile_Arn`** atau untuk meluncurkan instance baru.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Untuk detail API, lihat AddRoleToInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AddUserToGroup`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AddUserToGroup`.

CLI

AWS CLI

Untuk menambahkan pengguna ke grup IAM

`add-user-to-group`Perintah berikut menambahkan pengguna IAM bernama `Bob` ke grup IAM bernama. `Admins`

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus pengguna dalam grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat AddUserToGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini menambahkan nama pengguna **Bob** ke grup bernama**Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Untuk detail API, lihat AddUserToGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini menambahkan nama pengguna **Bob** ke grup bernama**Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Untuk detail API, lihat AddUserToGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AttachGroupPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachGroupPolicy`.

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke grup IAM

`attach-group-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `ReadOnlyAccess` ke grup IAM bernama. `Finance`

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat AttachGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan terkelola pelanggan yang diberi nama **TesterPolicy** ke grup IAM. **Testers** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi default kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini melampirkan kebijakan AWS terkelola bernama **AdministratorAccess** ke grup IAM. **Admins** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Untuk detail API, lihat AttachGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melampirkan kebijakan terkelola pelanggan yang diberi nama **TesterPolicy** ke grup IAM. **Testers** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi default kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini melampirkan kebijakan AWS terkelola bernama **`AdministratorAccess`** ke grup IAM. **`Admins`** Pengguna dalam grup tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Untuk detail API, lihat AttachGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AttachRolePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Konfigurasikan konektivitas layanan kontainer
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Buat API istirahat dengan integrasi proxy fungsi
- Membuat layanan kontainer untuk instance mesin virtual
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan orkestrasi alur kerja

- Kelola peran
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server
- Bekerja dengan Streams dan Time-to-Live

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
```

```
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }
```

```
        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke peran IAM

`attach-role-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `ReadOnlyAccess` ke peran IAM bernama. `ReadOnlyRole`

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
```

```
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Lampirkan kebijakan.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
      }
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
```

```
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Untuk detail API, lihat AttachRolePolicydi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
```

```
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola yang diberi nama **`SecurityAudit`** ke peran IAM. **`CoSecurityAuditors`** Pengguna yang menganggap peran tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Untuk detail API, lihat AttachRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola yang diberi nama **`SecurityAudit`** ke peran IAM. **`CoSecurityAuditors`** Pengguna yang menganggap peran tersebut langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Untuk detail API, lihat AttachRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Lampirkan kebijakan ke peran menggunakan objek Kebijakan Boto3.

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
```

```
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Lampirkan kebijakan ke peran menggunakan objek Peran Boto3.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

- Untuk detail API, lihat AttachRolePolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
 #{e.message}")
```

```
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
```

```
    false
  end
end
```

- Untuk detail API, lihat AttachRolePolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- Untuk detail API, lihat AttachRolePolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->attachrolepolicy(
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy attached to role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat AttachRolePolicydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func attachRolePolicy(role: String, policyArn: String) async throws {
        let input = AttachRolePolicyInput(
```

```
            policyArn: policyArn,
            roleName: role
        )
        do {
            _ = try await client.attachRolePolicy(input: input)
        } catch {
            print("ERROR: Attaching a role policy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi AttachRolePolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AttachUserPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AttachUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

CLI

AWS CLI

Untuk melampirkan kebijakan terkelola ke pengguna IAM

`attach-user-policy`Perintah berikut melampirkan kebijakan AWS terkelola bernama `AdministratorAccess` ke pengguna IAM bernama. `Alice`

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat AttachUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola bernama **`AmazonCognitoPowerUser`** ke pengguna IAM. **`Bob`** Pengguna langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Untuk detail API, lihat AttachUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melampirkan kebijakan AWS terkelola bernama **`AmazonCognitoPowerUser`** ke pengguna IAM. **`Bob`** Pengguna langsung terpengaruh oleh izin yang ditentukan dalam versi terbaru kebijakan tersebut.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Untuk detail API, lihat AttachUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Untuk detail API, lihat AttachUserPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Attaches a policy to a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_user(user_name, policy_arn)
    @iam_client.attach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to user: #{e.message}")
    false
  end
```

- Untuk detail API, lihat AttachUserPolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Untuk detail API, lihat AttachUserPolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    lo_iam->attachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat AttachUserPolicydi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ChangePassword`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ChangePassword`.

CLI

AWS CLI

Untuk mengubah kata sandi untuk pengguna IAM Anda

Untuk mengubah kata sandi untuk pengguna IAM Anda, sebaiknya gunakan `--cli-input-json` parameter untuk meneruskan file JSON yang berisi kata sandi lama dan baru Anda.

Dengan menggunakan metode ini, Anda dapat menggunakan kata sandi yang kuat dengan karakter non-alfanumerik. Mungkin sulit untuk menggunakan kata sandi dengan karakter non-alfanumerik ketika Anda meneruskannya sebagai parameter baris perintah. Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `change-password` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

Perintah sebelumnya membuat file JSON bernama change-password.json yang dapat Anda gunakan untuk mengisi kata sandi lama dan baru Anda. Misalnya, file mungkin terlihat seperti berikut ini.

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Selanjutnya, untuk mengubah kata sandi Anda, gunakan `change-password` perintah lagi, kali ini melewati `--cli-input-json` parameter untuk menentukan file JSON Anda. `change-password`Perintah berikut menggunakan `--cli-input-json` parameter dengan file JSON yang disebut change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Perintah ini tidak menghasilkan output.

Perintah ini hanya dapat dipanggil oleh pengguna IAM. Jika perintah ini disebut menggunakan kredensi AWS akun (root), perintah mengembalikan kesalahan`InvalidUserType`.

Untuk informasi selengkapnya, lihat Cara pengguna IAM mengubah kata sandi mereka sendiri di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ChangePassworddi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengubah kata sandi untuk pengguna yang menjalankan perintah. Perintah ini hanya dapat dipanggil oleh pengguna IAM. Jika perintah ini dipanggil ketika Anda masuk dengan kredensi AWS akun (root), perintah mengembalikan kesalahan. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Untuk detail API, lihat ChangePassworddi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mengubah kata sandi untuk pengguna yang menjalankan perintah. Perintah ini hanya dapat dipanggil oleh pengguna IAM. Jika perintah ini dipanggil ketika Anda masuk dengan kredensi AWS akun (root), perintah mengembalikan kesalahan. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Untuk detail API, lihat ChangePassworddi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateAccessKey`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write
- Kelola kunci akses

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
```

```
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk membuat kunci akses untuk pengguna IAM

`create-access-key`Perintah berikut membuat kunci akses (ID kunci akses dan kunci akses rahasia) untuk pengguna IAM bernamaBob.

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

Output:

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Simpan kunci akses rahasia di lokasi yang aman. Jika hilang, itu tidak dapat dipulihkan, dan Anda harus membuat kunci akses baru.

Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
```

```
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                   <user>\s
```

```
                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat kunci akses.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
```

```
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Untuk detail API, lihat CreateAccessKeydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kunci akses baru dan secret access key pair dan menetapkan ke pengguna**David**. Pastikan Anda menyimpan **AccessKeyId** dan **SecretAccessKey** nilai ke file karena ini adalah satu-satunya waktu Anda dapat memperoleh file**SecretAccessKey**. Anda tidak dapat mengambilnya di lain waktu. Jika Anda kehilangan kunci rahasia, Anda harus membuat access key pair baru.

```
New-IAMAccessKey -UserName David
```

Output:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Untuk detail API, lihat CreateAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kunci akses baru dan secret access key pair dan menetapkan ke pengguna**David**. Pastikan Anda menyimpan **AccessKeyId** dan **SecretAccessKey** nilai ke file karena ini adalah satu-satunya waktu Anda dapat memperoleh file**SecretAccessKey**. Anda tidak dapat mengambilnya di lain waktu. Jika Anda kehilangan kunci rahasia, Anda harus membuat access key pair baru.

```
New-IAMAccessKey -UserName David
```

Output:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

```
Status          : Active
UserName        : David
```

- Untuk detail API, lihat CreateAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Untuk detail API, lihat CreateAccessKeydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
```

```
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
```

```
    false
  end
end
```

- Untuk detail API, lihat CreateAccessKeydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Untuk detail API, lihat CreateAccessKeyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->createaccesskey(
      iv_username = iv_user_name ).
    MESSAGE 'Access key created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Maximum number of access keys reached.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat CreateAccessKeydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3
```

```
    public func createAccessKey(userName: String) async throws ->
IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            print("ERROR: createAccessKey:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi CreateAccessKey AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateAccountAlias`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateAccountAlias`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk membuat alias akun

`create-account-alias`Perintah berikut membuat alias `examplecorp` untuk AWS akun Anda.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
```

```
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat alias akun.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
```

```
    }
}
```

- Untuk detail API, lihat CreateAccountAliasdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah alias akun untuk AWS **mycompanyaws** akun Anda. <accountidnumber>Alamat halaman logon pengguna mengarah ke https://mycompanyaws.signin.aws.amazon.com/console. URL asli menggunakan nomor ID akun Anda, bukan alias (https://.signin.aws.amazon. com/console) terus bekerja. Namun, URL berbasis alias yang ditentukan sebelumnya berhenti berfungsi.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat CreateAccountAliasdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengubah alias akun untuk AWS **mycompanyaws** akun Anda. <accountidnumber>Alamat halaman logon pengguna mengarah ke https://mycompanyaws.signin.aws.amazon.com/console. URL asli menggunakan nomor ID akun Anda, bukan alias (https://.signin.aws.amazon. com/console) terus bekerja. Namun, URL berbasis alias yang ditentukan sebelumnya berhenti berfungsi.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat CreateAccountAliasdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- Untuk detail API, lihat CreateAccountAliasdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat CreateAccountAliasdi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->createaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Account alias already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Account alias limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat CreateAccountAliasdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateGroup`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateGroup`.

CLI

AWS CLI

Untuk membuat grup IAM

`create-group`Perintah berikut membuat grup IAM bernama`Admins`.

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

Output:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreateGroupdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat CreateGroupdi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat grup IAM baru bernama**`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Untuk detail API, lihat CreateGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat grup IAM baru bernama**`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Untuk detail API, lihat CreateGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateInstanceProfile`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membangun dan mengelola layanan yang tangguh
- Membuat layanan kontainer untuk instance mesin virtual
- Memulai dengan streaming terkelola
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
```

```
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                "\"Version\": \"2012-10-17\"," +
                                "\"Statement\": [{" +
                                     "\"Effect\": \"Allow\"," +
                                     "\"Principal\": {" +
                                     "\"Service\": [" +
                                         "\"ec2.amazonaws.com\"" +
                                     "]" +
                                     "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                           "}";

       var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

       var policyArn = "";

       try
       {
           var createPolicyResult = await _amazonIam.CreatePolicyAsync(
               new CreatePolicyRequest
               {
                   PolicyName = policyName,
                   PolicyDocument = policyDocument
               });
```

```
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
```

```
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Untuk detail API, lihat CreateInstanceProfiledi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk membuat profil instance

`create-instance-profile`Perintah berikut menciptakan sebuah contoh profil bernama`Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Output:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Untuk menambahkan peran ke profil instance, gunakan `add-role-to-instance-profile` perintah.

Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2 dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat CreateInstanceProfiledi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
```

- Untuk detail API, lihat CreateInstanceProfiledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat profil instans IAM baru bernama**`ProfileForDevEC2Instance`**. Anda harus menjalankan **`Add-IAMRoleToInstanceProfile`** perintah secara terpisah untuk mengaitkan profil instance dengan peran IAM yang ada yang memberikan izin ke instance. Terakhir, lampirkan profil instance ke instans EC2 saat Anda meluncurkannya. Untuk melakukan itu, gunakan **`New-EC2Instance`** cmdlet dengan parameter **`InstanceProfile_Arn`** or**`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Untuk detail API, lihat CreateInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat profil instans IAM baru bernama**`ProfileForDevEC2Instance`**. Anda harus menjalankan **`Add-IAMRoleToInstanceProfile`** perintah secara terpisah untuk mengaitkan profil instance dengan peran IAM yang ada yang memberikan izin ke instance. Terakhir, lampirkan profil instance ke instans EC2 saat Anda meluncurkannya. Untuk melakukan itu, gunakan **`New-EC2Instance`** cmdlet dengan parameter **`InstanceProfile_Arn`** or**`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Untuk detail API, lihat CreateInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Contoh ini membuat profil kebijakan, peran, dan instance dan menautkan semuanya.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
```

```
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
```

```
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
```

```
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

- Untuk detail API, lihat CreateInstanceProfiledi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateLoginProfile`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateLoginProfile`.

CLI

AWS CLI

Untuk membuat kata sandi untuk pengguna IAM

Untuk membuat kata sandi untuk pengguna IAM, sebaiknya gunakan `--cli-input-json` parameter untuk meneruskan file JSON yang berisi kata sandi. Dengan menggunakan metode ini, Anda dapat membuat kata sandi yang kuat dengan karakter non-alfanumerik. Mungkin sulit untuk membuat kata sandi dengan karakter non-alfanumerik ketika Anda meneruskannya sebagai parameter baris perintah.

Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `create-login-profile` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

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

Perintah sebelumnya membuat file JSON bernama create-login-profile.json yang dapat Anda gunakan untuk mengisi informasi untuk perintah berikutnya. `create-login-profile` Contoh:

```
{
```

```
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Selanjutnya, untuk membuat kata sandi untuk pengguna IAM, gunakan `create-login-profile` perintah lagi, kali ini melewati `--cli-input-json` parameter untuk menentukan file JSON Anda. `create-login-profile`Perintah berikut menggunakan `--cli-input-json` parameter dengan file JSON bernama create-login-profile.json.

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

Output:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Jika kata sandi baru melanggar kebijakan kata sandi akun, perintah mengembalikan `PasswordPolicyViolation` kesalahan.

Untuk mengubah kata sandi untuk pengguna yang sudah memilikinya, gunakan`update-login-profile`. Untuk menetapkan kebijakan kata sandi untuk akun, gunakan `update-account-password-policy` perintah.

Jika kebijakan kata sandi akun memungkinkan mereka, pengguna IAM dapat mengubah kata sandi mereka sendiri menggunakan `change-password` perintah.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreateLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kata sandi (sementara) untuk pengguna IAM bernama Bob, dan menetapkan tanda yang mengharuskan pengguna untuk mengubah kata sandi saat **Bob** masuk berikutnya.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Output:

```
CreateDate                  PasswordResetRequired             UserName
----------                  ---------------------             --------
4/14/2015 12:26:30 PM       True                              Bob
```

- Untuk detail API, lihat CreateLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kata sandi (sementara) untuk pengguna IAM bernama Bob, dan menetapkan tanda yang mengharuskan pengguna untuk mengubah kata sandi saat **Bob** masuk berikutnya.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Output:

```
CreateDate                  PasswordResetRequired             UserName
----------                  ---------------------             --------
4/14/2015 12:26:30 PM       True                              Bob
```

- Untuk detail API, lihat CreateLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateOpenIdConnectProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateOpenIdConnectProvider`.

CLI

AWS CLI

Untuk membuat penyedia OpenID Connect (OIDC)

Untuk membuat penyedia OpenID Connect (OIDC), sebaiknya gunakan `--cli-input-json` parameter untuk meneruskan file JSON yang berisi parameter yang diperlukan. Saat Anda membuat penyedia OIDC, Anda harus meneruskan URL penyedia, dan URL harus dimulai. `https://` Mungkin sulit untuk meneruskan URL sebagai parameter baris perintah, karena karakter titik dua (:) dan garis miring maju (/) memiliki arti khusus di beberapa lingkungan baris perintah. Menggunakan `--cli-input-json` parameter mengatasi batasan ini.

Untuk menggunakan `--cli-input-json` parameter, mulailah dengan menggunakan `create-open-id-connect-provider` perintah dengan `--generate-cli-skeleton` parameter, seperti pada contoh berikut.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

Perintah sebelumnya membuat file JSON bernama create-open-id-connect-provider.json yang dapat Anda gunakan untuk mengisi informasi untuk perintah berikutnya. `create-open-id-connect-provider` Contoh:

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Jika penemuan penyedia OIDC dan titik akhir JWKS Anda menggunakan sertifikat yang berbeda, sertakan cap jempol untuk kedua titik akhir di. Thumb`printList`

Selanjutnya, untuk membuat penyedia OpenID Connect (OIDC), gunakan `create-open-id-connect-provider` perintah lagi, kali ini melewati `--cli-input-json` parameter untuk menentukan file JSON Anda. `create-open-id-connect-provider`Perintah berikut menggunakan `--cli-input-json` parameter dengan file JSON bernama create-open-id-connect-provider.json.

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Output:

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Untuk informasi selengkapnya tentang penyedia OIDC, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

Untuk informasi selengkapnya tentang mendapatkan cap jempol untuk penyedia OIDC, lihat Memperoleh cap jempol untuk Penyedia Identitas OpenID Connect di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat CreateOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat penyedia IAM OIDC yang terkait dengan layanan penyedia kompatibel OIDC yang ditemukan di URL **`https://example.oidcprovider.com`** dan ID klien. **`my-testapp-1`** Penyedia OIDC memasok sidik jari. Untuk mengautentikasi sidik jari, ikuti langkah-langkah di http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Output:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat CreateOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat penyedia IAM OIDC yang terkait dengan layanan penyedia kompatibel OIDC yang ditemukan di URL **`https://example.oidcprovider.com`** dan ID klien. **`my-testapp-1`** Penyedia OIDC memasok sidik jari. Untuk mengautentikasi sidik jari, ikuti langkah-langkah di http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Output:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat CreateOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreatePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreatePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write
- Membuat ruang kerja pemantauan terkelola
- Memulai dengan streaming terkelola

- Memulai dengan orkestrasi alur kerja
- Kelola kebijakan
- Mengatur Kontrol Attribute-Based Akses
- Menyiapkan manajemen sistem
- Bekerja dengan API Pembuat Kebijakan IAM

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-policy operation failed.\n$response"
```

```
    return 1
  fi

  echo "$response"
}
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
```

```
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";

    return stringStream.str();
}
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Contoh 1: Untuk membuat kebijakan terkelola pelanggan

Perintah berikut membuat kebijakan terkelola pelanggan bernamamy-policy. File tersebut policy.json adalah dokumen JSON di folder saat ini yang memberikan akses baca saja ke shared folder dalam bucket Amazon S3 bernama. amzn-s3-demo-bucket

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json
```

Isi policy.json:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/shared/*"
            ]
        }
    ]
}
```

Output:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Untuk informasi lebih lanjut tentang menggunakan file sebagai input untuk parameter string, lihat Menentukan nilai parameter untuk AWS CLI di Panduan Pengguna AWS CLI.

Contoh 2: Untuk membuat kebijakan yang dikelola pelanggan dengan deskripsi

Perintah berikut membuat kebijakan terkelola pelanggan bernama `my-policy` dengan deskripsi yang tidak dapat diubah.

File tersebut `policy.json` adalah dokumen JSON di folder saat ini yang memberikan akses ke semua tindakan Put, List, dan Get untuk bucket Amazon S3 bernama. `amzn-s3-demo-bucket`

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for amzn-s3-demo-bucket"
```

Isi policy.json:

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Output:

```
{
    "Policy": {
```

```
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Untuk informasi selengkapnya tentang Idenity-based Kebijakan, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya di Panduan Pengguna IAM.AWS

Contoh 3: Untuk membuat kebijakan terkelola pelanggan dengan tag

Perintah berikut membuat kebijakan terkelola pelanggan bernama `my-policy` dengan tag. Contoh ini menggunakan `--tags` parameter dengan JSON-formatted tag berikut:`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Atau, `--tags` parameter dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

File tersebut `policy.json` adalah dokumen JSON di folder saat ini yang memberikan akses ke semua tindakan Put, List, dan Get untuk bucket Amazon S3 bernama. `amzn-s3-demo-bucket`

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Isi policy.json:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Output:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {
        ]
    }
}
```

Untuk informasi selengkapnya tentang kebijakan Penandaan, lihat Menandai kebijakan yang dikelola pelanggan di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreatePolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
```

```
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
```

```
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
+ result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);
```

```
            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat kebijakan.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
```

```
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
```

```
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "dynamodb:DeleteItem",
                    "dynamodb:GetItem",
                    "dynamodb:PutItem",
                    "dynamodb:Scan",
                    "dynamodb:UpdateItem"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Untuk detail API, lihat CreatePolicydi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    /**
     * @param string $policyName
     * @param string $policyDocument
     * @return array
     */
    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan IAM baru di AWS akun saat ini bernama **MySamplePolicy** File **MySamplePolicy.json** menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Untuk detail API, lihat CreatePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kebijakan IAM baru di AWS akun saat ini bernama **MySamplePolicy** File **MySamplePolicy.json** menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Untuk detail API, lihat CreatePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
```

```
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

- Untuk detail API, lihat CreatePolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
```

```
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
```

```
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat CreatePolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- Untuk detail API, lihat CreatePolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->createpolicy(
          iv_policyname = iv_policy_name
```

```
          iv_policydocument = iv_policy_document
          iv_description = iv_description ).
        MESSAGE 'Policy created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Policy already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat CreatePolicydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: createPolicy:", dump(error))
```

```
            throw error
        }
    }
```

- Untuk detail API, lihat referensi CreatePolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreatePolicyVersion`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreatePolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan

CLI

AWS CLI

Untuk membuat versi baru dari kebijakan terkelola

Contoh ini membuat `v2` versi baru dari kebijakan IAM yang `arn:aws:iam::123456789012:policy/MyPolicy` ARN-nya dan menjadikannya versi default.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Output:

```
{
    "PolicyVersion": {
```

```
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Untuk informasi selengkapnya, lihat Kebijakan IAM Versi di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreatePolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat versi “v2" baru dari kebijakan IAM yang ARN-nya **`arn:aws:iam::123456789012:policy/MyPolicy`** dan menjadikannya versi default. **`NewPolicyVersion.json`**File menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Output:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Untuk detail API, lihat CreatePolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat versi “v2" baru dari kebijakan IAM yang ARN-nya **`arn:aws:iam::123456789012:policy/MyPolicy`** dan menjadikannya versi default. **`NewPolicyVersion.json`**File menyediakan konten kebijakan. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses file kebijakan JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Output:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Untuk detail API, lihat CreatePolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
```

```
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

- Untuk detail API, lihat CreatePolicyVersiondi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->createpolicyversion(
          iv_policyarn = iv_policy_arn
          iv_policydocument = iv_policy_document
          iv_setasdefault = iv_set_as_default ).
        MESSAGE 'Policy version created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
```

```
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy version limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat CreatePolicyVersiondi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Konfigurasikan konektivitas layanan kontainer
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Buat API istirahat dengan integrasi proxy fungsi
- Membuat layanan kontainer untuk instance mesin virtual
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai dengan gudang data tanpa server
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Memulai dengan orkestrasi alur kerja

- Kelola peran
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server
- Bekerja dengan Streams dan Time-to-Live

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) role."
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat CreateRoledi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
```

```
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Contoh 1: Untuk membuat peran IAM

`create-role`Perintah berikut membuat peran bernama `Test-Role` dan melampirkan kebijakan kepercayaan padanya.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Output:

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

Kebijakan kepercayaan didefinisikan sebagai dokumen JSON dalam Test-Role-Trust-Policy.jsonfile. (Nama file dan ekstensi tidak memiliki signifikansi.) Kebijakan kepercayaan harus menentukan kepala sekolah.

Untuk melampirkan kebijakan izin ke peran, gunakan `put-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

Contoh 2: Untuk membuat peran IAM dengan durasi sesi maksimum yang ditentukan

`create-role`Perintah berikut membuat peran bernama `Test-Role` dan menetapkan durasi sesi maksimum 7200 detik (2 jam).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Untuk informasi selengkapnya, lihat Memodifikasi durasi sesi maksimum peran (AWS API) di Panduan Pengguna AWS IAM.

Contoh 3: Untuk membuat Peran IAM dengan tag

Perintah berikut membuat Peran IAM `Test-Role` dengan tag. Contoh ini menggunakan flag `--tags` parameter dengan JSON-formatted tag berikut:`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Atau, `--tags` bendera dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
```

```
            }
        ]
    }
}
```

Untuk informasi selengkapnya, lihat Menandai peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreateRoledi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
```

```
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s

                Where:
                    rolename - The name of the role to create.\s
```

```
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
```

```
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
          Action: "sts:AssumeRole",
        },
      ],
```

```
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
```

```
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat peran baru bernama **MyNewRole** dan melampirkan padanya kebijakan yang ditemukan dalam file**NewRoleTrustPolicy.json**. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON. Dokumen kebijakan yang ditampilkan dalam output adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **UrlDecode** metode.NET.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Output:

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                           %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                           %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                           %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                           %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
```

```
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Untuk detail API, lihat CreateRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat peran baru bernama **MyNewRole** dan melampirkan padanya kebijakan yang ditemukan dalam file**NewRoleTrustPolicy.json**. Perhatikan bahwa Anda harus menggunakan parameter **-Raw** switch untuk berhasil memproses file kebijakan JSON. Dokumen kebijakan yang ditampilkan dalam output adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **UrlDecode** metode.NET.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Output:

```
Arn                      : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
```

```
                            %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                            %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Untuk detail API, lihat CreateRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Untuk detail API, lihat CreateRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end

   role_arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating role: #{e.message}")
   nil
 end
```

- Untuk detail API, lihat CreateRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Untuk detail API, lihat CreateRolereferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->createrole(
      iv_rolename = iv_role_name
      iv_assumerolepolicydocument = iv_assume_role_policy_document ).
    MESSAGE 'Role created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamentityalrdyexex.
    MESSAGE 'Role already exists.' TYPE 'E'.
  CATCH /aws1/cx_iammalformedplydocex.
    MESSAGE 'Assume role policy document is malformed.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Role limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat CreateRoledi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3
```

```
    public func createRole(name: String, policyDocument: String) async throws ->
 String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            print("ERROR: createRole:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi CreateRole AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`createSamlProvider`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateSAMLProvider`.

CLI

AWS CLI

Untuk membuat penyedia SAFL

Contoh ini menciptakan penyedia SAFL baru di IAM bernama. `MySAMLProvider` Ini dijelaskan oleh dokumen metadata SALL yang ditemukan dalam file. `SAMLMetaData.xml`

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
```

```
    --name MySAMLProvider
```

Output:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas IAM SALL di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat CreateSamlProvider di Referensi Perintah.AWS CLI

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import * as path from "node:path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
```

```
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat CreateSamlProvider di Referensi API.AWS SDK untuk JavaScript

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat entitas penyedia SAFL baru di IAM. Ini dinamai **`MySAMLProvider`** dan dijelaskan oleh dokumen metadata SALL yang ditemukan dalam file**`SAMLMetaData.xml`**, yang diunduh secara terpisah dari situs web penyedia layanan SALL.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Output:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Untuk detail API, lihat CreateSamlProvider di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat entitas penyedia SAFL baru di IAM. Ini dinamai **`MySAMLProvider`** dan dijelaskan oleh dokumen metadata SALL yang ditemukan dalam file**`SAMLMetaData.xml`**, yang diunduh secara terpisah dari situs web penyedia layanan SALL.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Output:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Untuk detail API, lihat CreateSamlProvider di Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateServiceLinkedRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateServiceLinkedRole`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
```

```
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk membuat peran terkait layanan

`create-service-linked-role`Contoh berikut membuat peran terkait layanan untuk AWS layanan tertentu dan melampirkan deskripsi yang ditentukan.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Output:

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
```

```
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran terkait layanan.

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
```

```
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    }
    throw caught;
  }
};
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
```

```
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat peran servicelinked untuk layanan penskalaan otomatis.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat peran servicelinked untuk layanan penskalaan otomatis.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
    else:
        return role
```

- Untuk detail API, lihat CreateServiceLinkedRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
# Creates a service-linked role
#
# @param service_name [String] The service name to create the role for.
# @param description [String] The description of the service-linked role.
```

```
  # @param suffix [String] Suffix for customizing role name.
  # @return [String] The name of the created role
  def create_service_linked_role(service_name, description, suffix)
    response = @iam_client.create_service_linked_role(
      aws_service_name: service_name, description: description, custom_suffix:
 suffix
    )
    role_name = response.role.role_name
    @logger.info("Created service-linked role #{role_name}.")
    role_name
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't create service-linked role for #{service_name}.
 Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat CreateServiceLinkedRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
```

```
        .send()
        .await?;

    Ok(response)
}
```

- Untuk detail API, lihat CreateServiceLinkedRolereferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat CreateServiceLinkedRoledi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createServiceLinkedRole(service: String, suffix: String? = nil,
 description: String?)
                    async throws -> IAMClientTypes.Role {
        let input = CreateServiceLinkedRoleInput(
            awsServiceName: service,
            customSuffix: suffix,
            description: description
        )
        do {
            let output = try await client.createServiceLinkedRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: createServiceLinkedRole:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi CreateServiceLinkedRole AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateUser`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
```

```
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""
```

```
  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat CreateUserdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);
```

```
    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Contoh 1: Untuk membuat pengguna IAM

`create-user`Perintah berikut membuat pengguna IAM bernama Bob di akun saat ini.

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

Output:

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat pengguna IAM di AWS akun Anda di Panduan Pengguna AWS IAM.

Contoh 2: Untuk membuat pengguna IAM di jalur tertentu

`create-user`Perintah berikut membuat pengguna IAM bernama Bob di jalur yang ditentukan.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Output:

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat pengenal IAM di Panduan Pengguna AWS IAM.

Contoh 3: Untuk Membuat Pengguna IAM dengan tag

`create-user`Perintah berikut membuat pengguna IAM bernama Bob dengan tag. Contoh ini menggunakan flag `--tags` parameter dengan JSON-formatted tag berikut:`'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Atau, `--tags` bendera dapat digunakan dengan tag dalam format singkatan:. `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Output:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
```

```
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Untuk informasi selengkapnya, lihat Menandai pengguna IAM di Panduan Pengguna AWS IAM.

Contoh 3: Untuk membuat pengguna IAM dengan batas izin yang ditetapkan

`create-user`Perintah berikut membuat pengguna IAM bernama Bob dengan batas izin. AmazonS3FullAccess

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Output:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Untuk informasi selengkapnya, lihat Batasan izin untuk entitas IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat CreateUserdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
```

```
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
```

```
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();
```

```
            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Untuk detail API, lihat CreateUserdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat pengguna IAM bernama**Bob**. Jika Bob perlu masuk ke AWS konsol, maka Anda harus menjalankan perintah secara terpisah **`New-IAMLoginProfile`** untuk membuat profil masuk dengan kata sandi. Jika Bob perlu menjalankan AWS PowerShell atau perintah CLI lintas platform atau AWS melakukan panggilan API, maka Anda harus menjalankan **`New-IAMAccessKey`** perintah secara terpisah untuk membuat kunci akses.

```
New-IAMUser -UserName Bob
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
```

```
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Untuk detail API, lihat CreateUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat pengguna IAM bernama**Bob**. Jika Bob perlu masuk ke AWS konsol, maka Anda harus menjalankan perintah secara terpisah **`New-IAMLoginProfile`** untuk membuat profil masuk dengan kata sandi. Jika Bob perlu menjalankan AWS PowerShell atau perintah CLI lintas platform atau AWS melakukan panggilan API, maka Anda harus menjalankan **`New-IAMAccessKey`** perintah secara terpisah untuk membuat kunci akses.

```
New-IAMUser -UserName Bob
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Untuk detail API, lihat CreateUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def create_user(user_name):
    """
```

```
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Untuk detail API, lihat CreateUserdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
     user_name: user_name,
     password: initial_password,
     password_reset_required: true
```

```
    )
    @logger.info("User '#{user_name}' created successfully.")
    response.user.user_id
  rescue Aws::IAM::Errors::EntityAlreadyExists
    @logger.error("Error creating user '#{user_name}': user already exists.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating user '#{user_name}': #{e.message}")
    nil
  end
```

- Untuk detail API, lihat CreateUserdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Untuk detail API, lihat CreateUserreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->createuser(
          iv_username = iv_user_name ).
        MESSAGE 'User created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'User already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded for IAM users.' TYPE 'E'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Entity does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat CreateUserdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func createUser(name: String) async throws -> String {
```

```
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            print("ERROR: createUser:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi CreateUser AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateVirtualMfaDevice`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateVirtualMfaDevice`.

CLI

AWS CLI

Untuk membuat perangkat MFA virtual

Contoh ini menciptakan perangkat MFA virtual baru yang disebut. `BobsMFADevice` Ini membuat file yang berisi informasi bootstrap yang disebut `QRCode.png` dan menempatkannya di `C:/` direktori. Metode bootstrap yang digunakan dalam contoh ini adalah`QRCodePNG`.

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
```

```
    --bootstrap-method QRCodePNG
```

Output:

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multifaktor (MFA) dalam AWS dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat CreateVirtualMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **Base32StringSeed** nilai yang dibutuhkan program perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif dari kode QR). Setelah Anda mengkonfigurasi program dengan nilai, dapatkan dua kode otentikasi berurutan dari program. Terakhir, gunakan perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM **Bob** dan menyinkronkan akun dengan dua kode otentikasi.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Output:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Contoh 2: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **QRCodePNG** nilai dan menuliskannya ke file. Gambar ini dapat dipindai oleh program

perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif untuk memasukkan Base32StringSeed nilai secara manual). Setelah Anda membuat akun di program MFA virtual Anda, dapatkan dua kode otentikasi berurutan dan masukkan dalam perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM dan menyinkronkan akun. **Bob**

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Output:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Untuk detail API, lihat CreateVirtualMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **`Base32StringSeed`** nilai yang dibutuhkan program perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif dari kode QR). Setelah Anda mengkonfigurasi program dengan nilai, dapatkan dua kode otentikasi berurutan dari program. Terakhir, gunakan perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM **Bob** dan menyinkronkan akun dengan dua kode otentikasi.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Output:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --
```

```
Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Contoh 2: Contoh ini membuat perangkat MFA virtual baru. Baris 2 dan 3 mengekstrak **`QRCodePNG`** nilai dan menuliskannya ke file. Gambar ini dapat dipindai oleh program perangkat lunak MFA virtual untuk membuat akun (sebagai alternatif untuk memasukkan Base32StringSeed nilai secara manual). Setelah Anda membuat akun di program MFA virtual Anda, dapatkan dua kode otentikasi berurutan dan masukkan dalam perintah terakhir untuk menautkan perangkat MFA virtual ke pengguna IAM dan menyinkronkan akun. **`Bob`**

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Output:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Untuk detail API, lihat CreateVirtualMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeactivateMfaDevice`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeactivateMfaDevice`.

CLI

AWS CLI

Untuk menonaktifkan perangkat MFA

Perintah ini menonaktifkan perangkat MFA virtual dengan ARN `arn:aws:iam::210987654321:mfa/BobsMFADevice` yang terkait dengan pengguna. `Bob`

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multifaktor (MFA) dalam AWS dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeactivateMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini menonaktifkan perangkat MFA perangkat keras yang terkait dengan **Bob** pengguna yang memiliki nomor seri. **123456789012**

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Contoh 2: Perintah ini menonaktifkan perangkat MFA virtual yang terkait dengan **David** pengguna yang memiliki ARN. **arn:aws:iam::210987654321:mfa/David** Perhatikan bahwa perangkat MFA virtual tidak dihapus dari akun. Perangkat virtual masih ada dan muncul di output **Get-IAMVirtualMFADevice** perintah. Sebelum Anda dapat membuat perangkat MFA virtual baru untuk pengguna yang sama, Anda harus menghapus yang lama dengan menggunakan perintah. **Remove-IAMVirtualMFADevice**

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Untuk detail API, lihat DeactivateMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini menonaktifkan perangkat MFA perangkat keras yang terkait dengan **Bob** pengguna yang memiliki nomor seri. **123456789012**

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Contoh 2: Perintah ini menonaktifkan perangkat MFA virtual yang terkait dengan **David** pengguna yang memiliki ARN. **arn:aws:iam::210987654321:mfa/David** Perhatikan bahwa perangkat MFA virtual tidak dihapus dari akun. Perangkat virtual masih ada dan muncul di output **Get-IAMVirtualMFADevice** perintah. Sebelum Anda dapat membuat perangkat MFA virtual baru untuk pengguna yang sama, Anda harus menghapus yang lama dengan menggunakan perintah. **Remove-IAMVirtualMFADevice**

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Untuk detail API, lihat DeactivateMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DeleteAccessKey**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write
- Kelola kunci akses

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
```

```
}

##############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
```

```
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk menghapus kunci akses untuk pengguna IAM

`delete-access-key`Perintah berikut menghapus kunci akses yang ditentukan (ID kunci akses dan kunci akses rahasia) untuk pengguna IAM bernama. Bob

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk membuat daftar kunci akses yang ditentukan untuk pengguna IAM, gunakan `list-access-keys` perintah.

Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
```

```
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
```

```
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
```

```
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus tombol akses.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
```

```
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String,
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Untuk detail API, lihat DeleteAccessKeydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus AWS access key pair dengan ID kunci **AKIAIOSFODNN7EXAMPLE** dari nama **Bob** pengguna.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus AWS access key pair dengan ID kunci **AKIAIOSFODNN7EXAMPLE** dari nama **Bob** pengguna.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Untuk detail API, lihat DeleteAccessKeydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
```

```
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
```

```
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat DeleteAccessKeydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
```

```
    Ok(())
}
```

- Untuk detail API, lihat DeleteAccessKeyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    lo_iam->deleteaccesskey(
      iv_accesskeyid = iv_access_key_id
      iv_username = iv_user_name ).
    MESSAGE 'Access key deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Access key or user does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat DeleteAccessKeydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
```

```
import AWSS3


    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws
    {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            print("ERROR: deleteAccessKey:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeleteAccessKey AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteAccountAlias`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccountAlias`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk menghapus alias akun

`delete-account-alias`Perintah berikut menghapus alias `mycompany` untuk akun saat ini.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s
```

```
                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus alias akun.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
```

```
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Untuk detail API, lihat DeleteAccountAliasdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus alias akun dari akun Anda Akun AWS. Halaman masuk pengguna dengan alias di https://mycompanyaws.signin.aws.amazon.com/console tidak lagi berfungsi. Anda harus menggunakan URL asli dengan nomor Akun AWS ID Anda di https://<accountidnumber>.signin.aws.amazon. com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat DeleteAccountAliasdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus alias akun dari akun Anda Akun AWS. Halaman masuk pengguna dengan alias di https://mycompanyaws.signin.aws.amazon.com/console tidak lagi berfungsi. Anda harus menggunakan URL asli dengan nomor Akun AWS ID Anda di https://<accountidnumber>.signin.aws.amazon. com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Untuk detail API, lihat DeleteAccountAliasdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Untuk detail API, lihat DeleteAccountAliasdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
```

```
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat DeleteAccountAliasdi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    lo_iam->deleteaccountalias(
      iv_accountalias = iv_account_alias ).
    MESSAGE 'Account alias deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Account alias does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat DeleteAccountAliasdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteAccountPasswordPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Untuk menghapus kebijakan kata sandi akun saat ini

`delete-account-password-policy`Perintah berikut menghapus kebijakan kata sandi untuk akun saat ini.

```
aws iam delete-account-password-policy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteAccountPasswordPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan kata sandi untuk Akun AWS dan mengatur ulang semua nilai ke default aslinya. Jika kebijakan kata sandi saat ini tidak ada, pesan galat berikut akan muncul: Kebijakan akun dengan nama PasswordPolicy tidak dapat ditemukan.

```
Remove-IAMAccountPasswordPolicy
```

- Untuk detail API, lihat DeleteAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus kebijakan kata sandi untuk Akun AWS dan mengatur ulang semua nilai ke default aslinya. Jika kebijakan kata sandi saat ini tidak ada, pesan galat berikut akan muncul: Kebijakan akun dengan nama PasswordPolicy tidak dapat ditemukan.

```
Remove-IAMAccountPasswordPolicy
```

- Untuk detail API, lihat DeleteAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteGroup`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteGroup`.

CLI

AWS CLI

Untuk menghapus grup IAM

`delete-group`Perintah berikut menghapus grup IAM bernama. `MyTestGroup`

```
aws iam delete-group \
    --group-name MyTestGroup
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menghapus grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteGroupdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });
```

```
  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat DeleteGroupdi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus grup IAM bernama. **`MyTestGroup`** Perintah pertama menghapus setiap pengguna IAM yang merupakan anggota grup, dan perintah kedua menghapus grup IAM. Kedua perintah bekerja tanpa ada petunjuk untuk konfirmasi.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Untuk detail API, lihat DeleteGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus grup IAM bernama. **`MyTestGroup`** Perintah pertama menghapus setiap pengguna IAM yang merupakan anggota grup, dan perintah kedua menghapus grup IAM. Kedua perintah bekerja tanpa ada petunjuk untuk konfirmasi.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Untuk detail API, lihat DeleteGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteGroupPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteGroupPolicy`.

CLI

AWS CLI

Untuk menghapus kebijakan dari grup IAM

`delete-group-policy`Perintah berikut menghapus kebijakan yang dinamai `ExamplePolicy` dari grup bernama`Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk melihat kebijakan yang dilampirkan ke grup, gunakan `list-group-policies` perintah.

Untuk informasi selengkapnya, lihat Mengelola kebijakan IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`TesterPolicy`** dari grup IAM. **`Testers`** Pengguna dalam grup tersebut segera kehilangan izin yang ditentukan dalam kebijakan tersebut.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Untuk detail API, lihat DeleteGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`TesterPolicy`** dari grup IAM. **`Testers`** Pengguna dalam grup tersebut segera kehilangan izin yang ditentukan dalam kebijakan tersebut.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Untuk detail API, lihat DeleteGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteInstanceProfile`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membangun dan mengelola layanan yang tangguh
- Memulai dengan streaming terkelola
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
```

```
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Untuk detail API, lihat DeleteInstanceProfiledi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk menghapus profil instance

`delete-instance-profile`Perintah berikut menghapus profil instance bernama`ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Perintah ini tidak menghasilkan output.

Untuk informasi lebih lanjut, lihat Menggunakan profil instance dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeleteInstanceProfiledi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Untuk detail API, lihat DeleteInstanceProfiledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus profil instans EC2 bernama. **`MyAppInstanceProfile`** Perintah pertama melepaskan peran apa pun dari profil instance, dan kemudian perintah kedua menghapus profil instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Untuk detail API, lihat DeleteInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus profil instans EC2 bernama. **`MyAppInstanceProfile`** Perintah pertama melepaskan peran apa pun dari profil instance, dan kemudian perintah kedua menghapus profil instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Untuk detail API, lihat DeleteInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Contoh ini menghapus peran dari profil instance, melepaskan semua kebijakan yang dilampirkan pada peran, dan menghapus semua sumber daya.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
```

```
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
```

```
                    "Instance profile %s doesn't exist, nothing to do.",
 profile_name
                )
```

- Untuk detail API, lihat DeleteInstanceProfiledi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteLoginProfile`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteLoginProfile`.

CLI

AWS CLI

Untuk menghapus kata sandi untuk pengguna IAM

`delete-login-profile`Perintah berikut menghapus kata sandi untuk pengguna IAM bernama. `Bob`

```
aws iam delete-login-profile \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus profil login dari pengguna IAM bernama. **`Bob`** Ini mencegah pengguna masuk ke konsol. AWS Itu tidak mencegah pengguna menjalankan AWS CLI,

PowerShell, atau panggilan API apa pun menggunakan kunci AWS akses yang mungkin masih dilampirkan ke akun pengguna.

```
Remove-IAMLoginProfile -UserName Bob
```

- Untuk detail API, lihat DeleteLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus profil login dari pengguna IAM bernama. **Bob** Ini mencegah pengguna masuk ke konsol. AWS Itu tidak mencegah pengguna menjalankan AWS CLI, PowerShell, atau panggilan API apa pun menggunakan kunci AWS akses yang mungkin masih dilampirkan ke akun pengguna.

```
Remove-IAMLoginProfile -UserName Bob
```

- Untuk detail API, lihat DeleteLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteOpenIdConnectProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menghapus penyedia identitas OpenID Connect IAM

Contoh ini menghapus penyedia IAM OIDC yang terhubung ke penyedia. `example.oidcprovider.com`

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat DeleteOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyedia IAM OIDC yang terhubung ke penyedia. **`example.oidcprovider.com`** Pastikan Anda memperbarui atau menghapus peran apa pun yang mereferensikan penyedia ini dalam **`Principal`** elemen kebijakan kepercayaan peran.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat DeleteOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus penyedia IAM OIDC yang terhubung ke penyedia. **`example.oidcprovider.com`** Pastikan Anda memperbarui atau menghapus peran apa pun yang mereferensikan penyedia ini dalam **`Principal`** elemen kebijakan kepercayaan peran.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Untuk detail API, lihat DeleteOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeletePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeletePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write
- Membuat ruang kerja pemantauan terkelola
- Memulai dengan streaming terkelola
- Memulai dengan orkestrasi alur kerja
- Kelola kebijakan
- Menyiapkan manajemen sistem

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
################################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk menghapus kebijakan IAM

Contoh ini menghapus kebijakan `arn:aws:iam::123456789012:policy/MySamplePolicy` ARN-nya.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeletePolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
```

```
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
```

```
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus kebijakan.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Untuk detail API, lihat DeletePolicydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Untuk detail API, lihat DeletePolicydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan **`arn:aws:iam::123456789012:policy/MySamplePolicy`** ARN-nya. Sebelum Anda dapat menghapus kebijakan, Anda harus terlebih dahulu menghapus semua versi kecuali default dengan menjalankan**`Remove-IAMPolicyVersion`**. Anda juga harus melepaskan kebijakan dari setiap pengguna, grup, atau peran IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default, melepaskannya dari semua entitas IAM terlampir, dan akhirnya menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai versi default ke dalam koleksi dan kemudian menghapus setiap kebijakan dalam koleksi. Baris ketiga mengambil semua pengguna, grup, dan peran IAM yang dilampirkan kebijakan tersebut. Baris empat hingga enam melepaskan kebijakan dari setiap entitas terlampir. Baris terakhir menggunakan perintah ini untuk menghapus kebijakan terkelola serta versi default yang tersisa. Contohnya termasuk parameter **`-Force`** sakelar pada baris apa pun yang membutuhkannya untuk menekan permintaan konfirmasi.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Untuk detail API, lihat DeletePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus kebijakan **`arn:aws:iam::123456789012:policy/MySamplePolicy`** ARN-nya. Sebelum Anda dapat menghapus kebijakan, Anda harus terlebih dahulu menghapus semua versi kecuali default dengan menjalankan**`Remove-IAMPolicyVersion`**. Anda juga harus melepaskan kebijakan dari setiap pengguna, grup, atau peran IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default, melepaskannya dari semua entitas IAM terlampir, dan akhirnya menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai versi default ke dalam koleksi dan kemudian menghapus setiap kebijakan dalam koleksi. Baris ketiga mengambil semua

pengguna, grup, dan peran IAM yang dilampirkan kebijakan tersebut. Baris empat hingga enam melepaskan kebijakan dari setiap entitas terlampir. Baris terakhir menggunakan perintah ini untuk menghapus kebijakan terkelola serta versi default yang tersisa. Contohnya termasuk parameter **`-Force`** sakelar pada baris apa pun yang membutuhkannya untuk menekan permintaan konfirmasi.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Untuk detail API, lihat DeletePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Untuk detail API, lihat DeletePolicydi AWS SDK for Python (Boto3) Referensi API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Untuk detail API, lihat DeletePolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
```

```
        lo_iam->deletepolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Policy cannot be deleted due to attachments.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeletePolicydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            print("ERROR: deletePolicy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeletePolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeletePolicyVersion`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeletePolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk menghapus versi kebijakan terkelola

Contoh ini menghapus versi yang diidentifikasi sebagai `v2` dari kebijakan yang `arn:aws:iam::123456789012:policy/MySamplePolicy` ARN-nya.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeletePolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus versi yang diidentifikasi sebagai **`v2`** dari kebijakan yang **`arn:aws:iam::123456789012:policy/MySamplePolicy`** ARN-nya.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default dan kemudian menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai default ke dalam koleksi dan kemudian menggunakan perintah ini untuk menghapus setiap kebijakan dalam koleksi. Baris terakhir menghapus kebijakan itu sendiri serta versi default yang tersisa. Perhatikan bahwa agar berhasil menghapus kebijakan terkelola, Anda juga harus melepaskan kebijakan dari pengguna, grup, atau peran apa pun dengan menggunakan perintah**Unregister-IAMUserPolicy**,**Unregister-IAMGroupPolicy**, dan **Unregister-IAMRolePolicy** perintah. Lihat contoh untuk **Remove-IAMPolicy** cmdlet.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Untuk detail API, lihat DeletePolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus versi yang diidentifikasi sebagai **v2** dari kebijakan yang **arn:aws:iam::123456789012:policy/MySamplePolicy** ARN-nya.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Contoh 2: Contoh ini menghapus kebijakan dengan terlebih dahulu menghapus semua versi kebijakan non-default dan kemudian menghapus kebijakan itu sendiri. Baris pertama mengambil objek kebijakan. Baris kedua mengambil semua versi kebijakan yang tidak ditandai sebagai default ke dalam koleksi dan kemudian menggunakan perintah ini untuk menghapus setiap kebijakan dalam koleksi. Baris terakhir menghapus kebijakan itu sendiri serta versi default yang tersisa. Perhatikan bahwa agar berhasil menghapus kebijakan terkelola, Anda juga harus melepaskan kebijakan dari pengguna, grup, atau peran apa pun dengan menggunakan perintah**Unregister-IAMUserPolicy**,**Unregister-IAMGroupPolicy**,

dan **`Unregister-IAMRolePolicy`** perintah. Lihat contoh untuk **`Remove-IAMPolicy`** cmdlet.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Untuk detail API, lihat DeletePolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->deletepolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Policy version deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Cannot delete default policy version.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeletePolicyVersiondi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Konfigurasikan konektivitas layanan kontainer
- Buat API istirahat dengan integrasi proxy fungsi
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai dengan gudang data tanpa server
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Memulai dengan orkestrasi alur kerja
- Kelola peran
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteRoledi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
```

```
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
```

```
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat DeleteRoledi Referensi AWS CLI Perintah.

CLI

AWS CLI

Untuk menghapus peran IAM

`delete-role`Perintah berikut menghapus peran bernama`Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Perintah ini tidak menghasilkan output.

Sebelum dapat menghapus peran, Anda harus menghapus peran tersebut dari profil instance (`remove-role-from-instance-profile`), melepaskan kebijakan terkelola (`detach-role-policy`), dan menghapus kebijakan sebaris apa pun yang dilampirkan ke role (`delete-role-policy`).

Untuk informasi selengkapnya, lihat Membuat peran IAM dan Menggunakan profil instans di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteRoledi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
```

```
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Untuk detail API, lihat DeleteRoledi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus peran.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Untuk detail API, lihat DeleteRoledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran yang dinamai **MyNewRole** dari akun IAM saat ini. Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menggunakan **Unregister-IAMRolePolicy** perintah untuk melepaskan kebijakan terkelola apa pun. Kebijakan sebaris dihapus dengan peran.

```
Remove-IAMRole -RoleName MyNewRole
```

Contoh 2: Contoh ini melepaskan kebijakan terkelola dari peran bernama **MyNewRole** dan kemudian menghapus peran tersebut. Baris pertama mengambil kebijakan terkelola yang melekat pada peran sebagai koleksi dan kemudian melepaskan setiap kebijakan dalam koleksi dari peran tersebut. Baris kedua menghapus peran itu sendiri. Kebijakan inline dihapus bersama dengan peran.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Untuk detail API, lihat DeleteRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus peran yang dinamai **MyNewRole** dari akun IAM saat ini. Sebelum Anda dapat menghapus peran, Anda harus terlebih dahulu menggunakan

`Unregister-IAMRolePolicy` perintah untuk melepaskan kebijakan terkelola apa pun. Kebijakan sebaris dihapus dengan peran.

```
Remove-IAMRole -RoleName MyNewRole
```

Contoh 2: Contoh ini melepaskan kebijakan terkelola dari peran bernama **`MyNewRole`** dan kemudian menghapus peran tersebut. Baris pertama mengambil kebijakan terkelola yang melekat pada peran sebagai koleksi dan kemudian melepaskan setiap kebijakan dalam koleksi dari peran tersebut. Baris kedua menghapus peran itu sendiri. Kebijakan inline dihapus bersama dengan peran.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Untuk detail API, lihat DeleteRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Untuk detail API, lihat DeleteRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
                                         role_name: role_name,
                                         policy_arn: policy.policy_arn
                                       })
        # Check if the policy is a customer managed policy (not AWS managed)
        unless policy.policy_arn.include?('aws:policy/')
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
        end
      end
    end

    # Delete the role
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
 why:")
```

```
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat DeleteRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Untuk detail API, lihat DeleteRolereferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->deleterole( iv_rolename = iv_role_name ).
        MESSAGE 'Role deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Role cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeleteRoledi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
```

```
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            print("ERROR: deleteRole:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeleteRole AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteRolePermissionsBoundary`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Untuk menghapus batas izin dari peran IAM

`delete-role-permissions-boundary`Contoh berikut menghapus batas izin untuk peran IAM yang ditentukan. Untuk menerapkan batas izin ke peran, gunakan perintah. `put-role-permissions-boundary`

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeleteRolePermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke peran IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Untuk detail API, lihat DeleteRolePermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke peran IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Untuk detail API, lihat DeleteRolePermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteRolePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan gudang data tanpa server
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan klaster gudang data yang disediakan
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteRolePolicydi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk menghapus kebijakan dari peran IAM

`delete-role-policy`Perintah berikut menghapus kebijakan yang dinamai `ExamplePolicy` dari peran yang dinamai`Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteRolePolicydi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Untuk detail API, lihat DeleteRolePolicydi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline **`S3AccessPolicy`** yang disematkan dalam peran IAM. **`S3BackupRole`**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Untuk detail API, lihat DeleteRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus kebijakan inline **`S3AccessPolicy`** yang disematkan dalam peran IAM. **`S3BackupRole`**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Untuk detail API, lihat DeleteRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteSamlProvider`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteSAMLProvider`.

CLI

AWS CLI

Untuk menghapus penyedia SAFL

Contoh ini menghapus penyedia IAM SALL 2.0 yang ARN-nya. `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas IAM SALL di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteSamlProvider di Referensi Perintah.AWS CLI

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat DeleteSamlProvider di Referensi API.AWS SDK untuk JavaScript

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus penyedia IAM SALL 2.0 yang ARN-nya. **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Untuk detail API, lihat DeleteSamlProvider di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus penyedia IAM SALL 2.0 yang ARN-nya. **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Untuk detail API, lihat DeleteSamlProvider di Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteServerCertificate`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteServerCertificate`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
   Aws::IAM::IAMClient iam(clientConfig);
   Aws::IAM::Model::DeleteServerCertificateRequest request;
   request.SetServerCertificateName(certificateName);

   const auto outcome = iam.DeleteServerCertificate(request);
   bool result = true;
   if (!outcome.IsSuccess()) {
       if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
           std::cerr << "Error deleting server certificate " << certificateName
 <<
                     ": " << outcome.GetError().GetMessage() << std::endl;
           result = false;
       }
       else {
           std::cout << "Certificate '" << certificateName
                     << "' not found." << std::endl;
       }
   }
   else {
       std::cout << "Successfully deleted server certificate " <<
 certificateName
                 << std::endl;
   }

   return result;
}
```

- Untuk detail API, lihat DeleteServerCertificatedi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk menghapus sertifikat server dari AWS akun Anda

`delete-server-certificate`Perintah berikut menghapus sertifikat server yang ditentukan dari AWS akun Anda.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Perintah ini tidak menghasilkan output.

Untuk membuat daftar sertifikat server yang tersedia di AWS akun Anda, gunakan `list-server-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server di IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus sertifikat server.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteServerCertificatedi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteServerCertificatedi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus sertifikat server bernama**MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Untuk detail API, lihat DeleteServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus sertifikat server bernama**MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Untuk detail API, lihat DeleteServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
```

```
                                          certificate_body: certificate_body,
                                          private_key: private_key
                                        })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat DeleteServerCertificatedi Referensi AWS SDK untuk Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteServiceLinkedRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteServiceLinkedRole`.

CLI

AWS CLI

Untuk menghapus peran terkait layanan

`delete-service-linked-role`Contoh berikut menghapus peran terkait layanan yang ditentukan yang tidak lagi Anda perlukan. Penghapusan terjadi secara asinkron. Anda dapat memeriksa status penghapusan dan mengonfirmasi kapan dilakukan dengan menggunakan perintah. `get-service-linked-role-deletion-status`

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Output:

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
```

```
 }
 return err
}
```

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran terkait layanan. Harap dicatat bahwa jika layanan masih menggunakan peran ini, maka perintah ini mengakibatkan kegagalan.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus peran terkait layanan. Harap dicatat bahwa jika layanan masih menggunakan peran ini, maka perintah ini mengakibatkan kegagalan.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Deletes a service-linked role.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_service_linked_role(role_name)
    response = @iam_client.delete_service_linked_role(role_name: role_name)
```

```
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id
      )
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)

      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    return if e.code == 'NoSuchEntity'

    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat DeleteServiceLinkedRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Untuk detail API, lihat DeleteServiceLinkedRolereferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteSigningCertificate`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteSigningCertificate`.

CLI

AWS CLI

Untuk menghapus sertifikat penandatanganan untuk pengguna IAM

`delete-signing-certificate`Perintah berikut menghapus sertifikat penandatanganan yang ditentukan untuk pengguna IAM bernama. Bob

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Perintah ini tidak menghasilkan output.

Untuk mendapatkan ID untuk sertifikat penandatanganan, gunakan `list-signing-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan Pengguna Amazon EC2.

- Untuk detail API, lihat DeleteSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus sertifikat penandatanganan dengan ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** dari pengguna IAM bernama. **Bob**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Untuk detail API, lihat DeleteSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus sertifikat penandatanganan dengan ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** dari pengguna IAM bernama. **Bob**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Untuk detail API, lihat DeleteSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteUser`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Buat pengguna read-only dan read-write

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
########################################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
```

```
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
```

```
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk menghapus pengguna IAM

`delete-user`Perintah berikut menghapus nama pengguna IAM Bob dari akun saat ini.

```
aws iam delete-user \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menghapus pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteUserdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
```

```
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
```

```
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Hapus pengguna.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Untuk detail API, lihat DeleteUserdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus nama pengguna IAM. **Bob**

```
Remove-IAMUser -UserName Bob
```

Contoh 2: Contoh ini menghapus nama pengguna IAM **Theresa** bersama dengan elemen apa pun yang harus dihapus terlebih dahulu.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

```
# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Untuk detail API, lihat DeleteUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus nama pengguna IAM. **Bob**

```
Remove-IAMUser -UserName Bob
```

Contoh 2: Contoh ini menghapus nama pengguna IAM **Theresa** bersama dengan elemen apa pun yang harus dihapus terlebih dahulu.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
```

```
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Untuk detail API, lihat DeleteUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- Untuk detail API, lihat DeleteUserdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Untuk detail API, lihat DeleteUserdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Untuk detail API, lihat DeleteUserreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->deleteuser( iv_username = iv_user_name ).
        MESSAGE 'User deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'User cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeleteUserdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func deleteUser(user: IAMClientTypes.User) async throws {
        let input = DeleteUserInput(
            userName: user.userName
        )
```

```
        do {
            _ = try await iamClient.deleteUser(input: input)
        } catch {
            print("ERROR: deleteUser:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeleteUser AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteUserPermissionsBoundary`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Untuk menghapus batas izin dari pengguna IAM

`delete-user-permissions-boundary`Contoh berikut menghapus batas izin yang dilampirkan ke pengguna IAM bernama. `intern` Untuk menerapkan batas izin ke pengguna, gunakan perintah. `put-user-permissions-boundary`

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat DeleteUserPermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke pengguna IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Untuk detail API, lihat DeleteUserPermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menunjukkan cara menghapus batas izin yang dilampirkan ke pengguna IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Untuk detail API, lihat DeleteUserPermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteUserPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteUserPolicydi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk menghapus kebijakan dari pengguna IAM

`delete-user-policy`Perintah berikut menghapus kebijakan yang ditentukan dari pengguna IAM bernamaBob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Perintah ini tidak menghasilkan output.

Untuk mendapatkan daftar kebijakan untuk pengguna IAM, gunakan `list-user-policies` perintah.

Untuk informasi selengkapnya, lihat Membuat pengguna IAM di AWS akun Anda di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DeleteUserPolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
```

```
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Untuk detail API, lihat DeleteUserPolicydi Referensi AWS SDK untuk Go API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`AccessToEC2Policy`** yang disematkan dalam nama pengguna IAM. **`Bob`**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Contoh 2: Contoh ini menemukan semua kebijakan inline yang disematkan dalam nama pengguna IAM **`Theresa`** dan kemudian menghapusnya.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Untuk detail API, lihat DeleteUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus kebijakan inline bernama **`AccessToEC2Policy`** yang disematkan dalam nama pengguna IAM. **`Bob`**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Contoh 2: Contoh ini menemukan semua kebijakan inline yang disematkan dalam nama pengguna IAM **`Theresa`** dan kemudian menghapusnya.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Untuk detail API, lihat DeleteUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Untuk detail API, lihat DeleteUserPolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Untuk detail API, lihat DeleteUserPolicyreferensi AWS SDK for Rust API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
 throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            print("ERROR: deleteUserPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeleteUserPolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteVirtualMfaDevice`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteVirtualMfaDevice`.

CLI

AWS CLI

Untuk menghapus perangkat MFA virtual

`delete-virtual-mfa-device`Perintah berikut menghapus perangkat MFA yang ditentukan dari akun saat ini.

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menonaktifkan perangkat MFA di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat DeleteVirtualMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus perangkat MFA virtual IAM yang ARN-nya. **arn:aws:iam::123456789012:mfa/bob**

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Contoh 2: Contoh ini memeriksa untuk melihat apakah pengguna IAM Theresa memiliki perangkat MFA yang ditetapkan. Jika ditemukan, perangkat dinonaktifkan untuk pengguna IAM. Jika perangkat virtual, maka itu juga dihapus.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Untuk detail API, lihat DeleteVirtualMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus perangkat MFA virtual IAM yang ARN-nya. **arn:aws:iam::123456789012:mfa/bob**

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Contoh 2: Contoh ini memeriksa untuk melihat apakah pengguna IAM Theresa memiliki perangkat MFA yang ditetapkan. Jika ditemukan, perangkat dinonaktifkan untuk pengguna IAM. Jika perangkat virtual, maka itu juga dihapus.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Untuk detail API, lihat DeleteVirtualMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DetachGroupPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachGroupPolicy`.

CLI

AWS CLI

Untuk melepaskan kebijakan dari grup

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` dari grup yang dipanggil. `Testers`

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DetachGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** ARNnya berasal dari grup bernama. **`Testers`**

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada grup bernama **`Testers`** dan memisahkannya dari grup.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Untuk detail API, lihat DetachGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** ARNnya berasal dari grup bernama. **`Testers`**

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada grup bernama **`Testers`** dan memisahkannya dari grup.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Untuk detail API, lihat DetachGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DetachRolePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya
- Konfigurasikan konektivitas layanan kontainer
- Buat API istirahat dengan integrasi proxy fungsi
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan orkestrasi alur kerja
- Kelola peran
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk melepaskan kebijakan dari peran

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` dari peran yang dipanggil. `FedTesterRole`

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
```

```
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
```

```
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
```

```
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Lepaskan kebijakan.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
```

```
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
```

```
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
```

```
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Untuk detail API, lihat DetachRolePolicydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang ARNnya berasal dari peran yang **arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy** dinamai. **FedTesterRole**

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada peran yang diberi nama **FedTesterRole** dan memisahkannya dari peran.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Untuk detail API, lihat DetachRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melepaskan kebijakan grup terkelola yang ARNnya berasal dari peran yang **arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy** dinamai. **FedTesterRole**

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan pada peran yang diberi nama **FedTesterRole** dan memisahkannya dari peran.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Untuk detail API, lihat DetachRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Lepaskan kebijakan dari peran menggunakan objek Kebijakan Boto3.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Lepaskan kebijakan dari peran menggunakan objek Peran Boto3.

```
def detach_policy(role_name, policy_arn):
```

```
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Untuk detail API, lihat DetachRolePolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
```

```
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
```

```
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat DetachRolePolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Untuk detail API, lihat DetachRolePolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->detachrolepolicy(
```

```
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DetachRolePolicydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
 IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
            policyArn: policy.arn,
            roleName: role.roleName
        )

        do {
            _ = try await iamClient.detachRolePolicy(input: input)
        } catch {
            print("ERROR: detachRolePolicy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DetachRolePolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DetachUserPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DetachUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

CLI

AWS CLI

Untuk melepaskan kebijakan dari pengguna

Contoh ini menghapus kebijakan terkelola dengan ARN `arn:aws:iam::123456789012:policy/TesterPolicy` dari pengguna. Bob

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengubah izin untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DetachUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini melepaskan kebijakan terkelola yang ARNnya berasal dari **`arn:aws:iam::123456789012:policy/TesterPolicy`** nama pengguna IAM. **`Bob`**

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan ke pengguna IAM bernama **`Theresa`** dan melepaskan kebijakan tersebut dari pengguna.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Untuk detail API, lihat DetachUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini melepaskan kebijakan terkelola yang ARNnya berasal dari **`arn:aws:iam::123456789012:policy/TesterPolicy`** nama pengguna IAM. **`Bob`**

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Contoh 2: Contoh ini menemukan semua kebijakan terkelola yang dilampirkan ke pengguna IAM bernama **`Theresa`** dan melepaskan kebijakan tersebut dari pengguna.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Untuk detail API, lihat DetachUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Untuk detail API, lihat DetachUserPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
```

```
    )
    @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
 successfully.")
    true
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error('Error detaching policy: Policy or user does not exist.')
    false
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from user '#{user_name}':
 #{e.message}")
    false
  end
```

- Untuk detail API, lihat DetachUserPolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Untuk detail API, lihat DetachUserPolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    lo_iam->detachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy detached from user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat DetachUserPolicydi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`EnableMfaDevice`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`EnableMfaDevice`.

CLI

AWS CLI

Untuk mengaktifkan perangkat MFA

Setelah Anda menggunakan `create-virtual-mfa-device` perintah untuk membuat perangkat MFA virtual baru, Anda dapat menetapkan perangkat MFA ke

pengguna. `enable-mfa-device`Contoh berikut menetapkan perangkat MFA dengan `arn:aws:iam::210987654321:mfa/BobsMFADevice` nomor seri kepada pengguna. Bob Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari perangkat MFA virtual.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengaktifkan perangkat otentikasi multi-faktor virtual (MFA) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat EnableMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini memungkinkan perangkat MFA perangkat keras dengan nomor seri **987654321098** dan mengaitkan perangkat dengan pengguna. **Bob** Ini termasuk dua kode pertama secara berurutan dari perangkat.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Contoh 2: Contoh ini membuat dan mengaktifkan perangkat MFA virtual. Perintah pertama membuat perangkat virtual dan mengembalikan representasi objek perangkat dalam variabel**$MFADevice**. Anda dapat menggunakan **QRCodePng** properti **.Base32StringSeed** atau untuk mengkonfigurasi aplikasi perangkat lunak pengguna. Perintah terakhir menetapkan perangkat kepada pengguna**David**, mengidentifikasi perangkat dengan nomor serinya. Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari perangkat MFA virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
```

```
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Untuk detail API, lihat EnableMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini memungkinkan perangkat MFA perangkat keras dengan nomor seri **987654321098** dan mengaitkan perangkat dengan pengguna. **Bob** Ini termasuk dua kode pertama secara berurutan dari perangkat.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Contoh 2: Contoh ini membuat dan mengaktifkan perangkat MFA virtual. Perintah pertama membuat perangkat virtual dan mengembalikan representasi objek perangkat dalam variabel**$MFADevice**. Anda dapat menggunakan **QRCodePng** properti **.Base32StringSeed** atau untuk mengkonfigurasi aplikasi perangkat lunak pengguna. Perintah terakhir menetapkan perangkat kepada pengguna**David**, mengidentifikasi perangkat dengan nomor serinya. Perintah ini juga menyinkronkan perangkat AWS dengan memasukkan dua kode pertama secara berurutan dari perangkat MFA virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Untuk detail API, lihat EnableMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **GenerateCredentialReport**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GenerateCredentialReport`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk menghasilkan laporan kredensi

Contoh berikut mencoba untuk menghasilkan laporan kredensi untuk AWS akun.

```
aws iam generate-credential-report
```

Output:

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Untuk informasi selengkapnya, lihat Mendapatkan laporan kredensi untuk AWS akun Anda di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GenerateCredentialReportdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini meminta pembuatan laporan baru, yang dapat dilakukan setiap empat jam. Jika laporan terakhir masih terbaru, bidang Negara berbunyi**`COMPLETE`**. Gunakan **`Get-IAMCredentialReport`** untuk melihat laporan yang sudah selesai.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                  State
-----------                                                  -----
```

```
No report exists. Starting a new report generation task        STARTED
```

- Untuk detail API, lihat GenerateCredentialReportdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini meminta pembuatan laporan baru, yang dapat dilakukan setiap empat jam. Jika laporan terakhir masih terbaru, bidang Negara berbunyi**COMPLETE**. Gunakan **Get-IAMCredentialReport** untuk melihat laporan yang sudah selesai.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Untuk detail API, lihat GenerateCredentialReportdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
```

```
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
%s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
account.")
        raise
    else:
        return response
```

- Untuk detail API, lihat GenerateCredentialReportdi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->generatecredentialreport( ).
        MESSAGE 'Credential report generation started.' TYPE 'I'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Report generation limit exceeded.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when generating credential report.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GenerateCredentialReportdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GenerateServiceLastAccessedDetails`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Contoh 1: Untuk membuat laporan akses layanan untuk kebijakan kustom

`generate-service-last-accessed-details`Contoh berikut memulai pekerjaan latar belakang untuk menghasilkan laporan yang mencantumkan layanan yang diakses oleh pengguna IAM dan entitas lain dengan kebijakan khusus bernama`intern-boundary`. Anda dapat menampilkan laporan setelah dibuat dengan menjalankan `get-service-last-accessed-details` perintah.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Output:

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Contoh 2: Untuk membuat laporan akses layanan untuk AdministratorAccess kebijakan AWS terkelola

`generate-service-last-accessed-details`Contoh berikut memulai pekerjaan latar belakang untuk menghasilkan laporan yang mencantumkan layanan yang diakses oleh pengguna IAM dan entitas lain dengan `AdministratorAccess` kebijakan AWS terkelola. Anda dapat menampilkan laporan setelah dibuat dengan menjalankan `get-service-last-accessed-details` perintah.

```
aws iam generate-service-last-accessed-details \
```

```
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Output:

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GenerateServiceLastAccessedDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini setara dengan cmdlet API. GenerateServiceLastAccessedDetails Ini menyediakan id pekerjaan yang dapat digunakan di Get-IAMServiceLastAccessedDetail dan Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Untuk detail API, lihat GenerateServiceLastAccessedDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini setara dengan cmdlet API. GenerateServiceLastAccessedDetails Ini menyediakan id pekerjaan yang dapat digunakan di Get-IAMServiceLastAccessedDetail dan Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Untuk detail API, lihat GenerateServiceLastAccessedDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetAccessKeyLastUsed`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccessKeyLastUsed`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);

    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
```

```
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengambil informasi tentang kapan kunci akses yang ditentukan terakhir digunakan

Contoh berikut mengambil informasi tentang kapan kunci akses ABCDEXAMPLE terakhir digunakan.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Output:

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kunci akses.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan nama pengguna yang dimiliki dan informasi penggunaan terakhir untuk kunci akses yang disediakan.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan nama pengguna yang dimiliki dan informasi penggunaan terakhir untuk kunci akses yang disediakan.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Untuk detail API, lihat GetAccessKeyLastUseddi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
```

```
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Untuk detail API, lihat GetAccessKeyLastUseddi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->getaccesskeylastused(
      iv_accesskeyid = iv_access_key_id ).
    MESSAGE 'Retrieved access key last used information.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Access key does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat GetAccessKeyLastUseddi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetAccountAuthorizationDetails`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountAuthorizationDetails`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk mencantumkan pengguna, grup, peran, dan kebijakan IAM AWS akun

`get-account-authorization-details`Perintah berikut menampilkan informasi tentang semua pengguna, grup, peran, dan kebijakan IAM di AWS akun.

```
aws iam get-account-authorization-details
```

Output:

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "RoleId": "AROA1234567890EXAMPLE",
            "CreateDate": "2014-07-30T17:09:20Z",
            "InstanceProfileList": [
                {
                    "InstanceProfileId": "AIPA1234567890EXAMPLE",
                    "Roles": [
                        {
                            "AssumeRolePolicyDocument": {
```

```
                                "Version":"2012-10-17",
                                "Statement": [
                                    {
                                        "Sid": "",
                                        "Effect": "Allow",
                                        "Principal": {
                                            "Service": "ec2.amazonaws.com"
                                        },
                                        "Action": "sts:AssumeRole"
                                    }
                                ]
                            },
                            "RoleId": "AROA1234567890EXAMPLE",
                            "CreateDate": "2014-07-30T17:09:20Z",
                            "RoleName": "EC2role",
                            "Path": "/",
                            "Arn": "arn:aws:iam::123456789012:role/EC2role"
                        }
                    ],
                    "CreateDate": "2014-07-30T17:09:20Z",
                    "InstanceProfileName": "EC2role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
                }
            ],
            "RoleName": "EC2role",
            "Path": "/",
            "AttachedManagedPolicies": [
                {
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
```

```
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
```

```
                            "Effect": "Allow"
                        }
                    ]
                }
            }
        ]
    }
],
"UserDetailList": [
    {
        "UserName": "Alice",
        "GroupList": [
            "Admins"
        ],
        "CreateDate": "2013-10-14T18:32:24Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Alice"
    },
    {
        "UserName": "Bob",
        "GroupList": [
            "Admins"
        ],
        "CreateDate": "2013-10-14T18:32:25Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [
            {
                "PolicyName": "DenyBillingAndIAMPolicy",
                "PolicyDocument": {
                    "Version":"2012-10-17",
                    "Statement": {
                        "Effect": "Deny",
                        "Action": [
                            "aws-portal:*",
                            "iam:*"
                        ],
                        "Resource": "*"
                    }
                }
            },
```

```
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
```

```
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::amzn-s3-demo-bucket",
                                    "arn:aws:s3:::amzn-s3-demo-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
```

```
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
```

```
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Untuk informasi selengkapnya, lihat pedoman audit AWS keamanan di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetAccountAuthorizationDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mendapatkan rincian otorisasi tentang identitas di AWS akun, dan menampilkan daftar elemen objek yang dikembalikan, termasuk pengguna, grup, dan peran. Misalnya, **`UserDetailList`** properti menampilkan detail tentang pengguna. Informasi serupa tersedia di **`RoleDetailList`** dan **`GroupDetailList`** properti.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Output:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Output:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
```

```
UserName       : Administrator
UserPolicyList : {}

Arn            : arn:aws:iam::123456789012:user/Bob
CreateDate     : 4/6/2015 12:54:42 PM
GroupList      : {Developers}
Path           : /
UserId         : AIDACKCEVSQ6CEXAMPLE2
UserName       : bab
UserPolicyList : {}

Arn            : arn:aws:iam::123456789012:user/BackupToS3
CreateDate     : 1/27/2015 10:15:08 AM
GroupList      : {Backup}
Path           : /
UserId         : AIDACKCEVSQ6CEXAMPLE3
UserName       : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Untuk detail API, lihat GetAccountAuthorizationDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mendapatkan rincian otorisasi tentang identitas di AWS akun, dan menampilkan daftar elemen objek yang dikembalikan, termasuk pengguna, grup, dan peran. Misalnya, **`UserDetailList`** properti menampilkan detail tentang pengguna. Informasi serupa tersedia di **`RoleDetailList`** dan **`GroupDetailList`** properti.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Output:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Output:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Untuk detail API, lihat GetAccountAuthorizationDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_authorization_details(response_filter):
    """
```

```
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Untuk detail API, lihat GetAccountAuthorizationDetailsdi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getaccountauthdetails( ).
        MESSAGE 'Retrieved account authorization details.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting account authorization details.'
TYPE 'E'.
```

```
    ENDTRY.
```

- Untuk detail API, lihat GetAccountAuthorizationDetailsdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetAccountPasswordPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountPasswordPolicy`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk melihat kebijakan kata sandi akun saat ini

`get-account-password-policy`Perintah berikut menampilkan detail tentang kebijakan kata sandi untuk akun saat ini.

```
aws iam get-account-password-policy
```

Output:

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan `NoSuchEntity` kesalahan.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kebijakan kata sandi akun.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan detail tentang kebijakan kata sandi untuk akun saat ini. Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan **`NoSuchEntity`** kesalahan.

```
Get-IAMAccountPasswordPolicy
```

Output:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan detail tentang kebijakan kata sandi untuk akun saat ini. Jika tidak ada kebijakan kata sandi yang ditentukan untuk akun, perintah mengembalikan **`NoSuchEntity`** kesalahan.

```
Get-IAMAccountPasswordPolicy
```

Output:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
```

```
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
{pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
{pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
{pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
{pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Untuk detail API, lihat GetAccountPasswordPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'IAMPolicyManager'
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    response = @iam_client.get_account_password_policy
    @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.info('The account does not have a password policy.')
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't print the account password policy. Error: #{e.code} -
 #{e.message}")
    raise
  end
end
```

- Untuk detail API, lihat GetAccountPasswordPolicydi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Untuk detail API, lihat GetAccountPasswordPolicyreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getaccountpasswordpolicy( ).
        MESSAGE 'Retrieved account password policy.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'No password policy exists.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting password policy.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetAccountPasswordPolicydi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetAccountSummary`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetAccountSummary`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk mendapatkan informasi tentang penggunaan entitas IAM dan kuota IAM di akun saat ini

`get-account-summary`Perintah berikut mengembalikan informasi tentang penggunaan entitas IAM saat ini dan kuota entitas IAM saat ini di akun.

```
aws iam get-account-summary
```

Output:

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
```

```
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Untuk informasi selengkapnya tentang batasan entitas, lihat kuota IAM dan AWS STS di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetAccountSummarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan informasi tentang penggunaan entitas IAM saat ini dan kuota entitas IAM saat ini di. Akun AWS

```
Get-IAMAccountSummary
```

Output:

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
```

```
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Untuk detail API, lihat GetAccountSummarydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan informasi tentang penggunaan entitas IAM saat ini dan kuota entitas IAM saat ini di. Akun AWS

```
Get-IAMAccountSummary
```

Output:

```

Key                                     Value
Users                                   7
```

```
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Untuk detail API, lihat GetAccountSummarydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Untuk detail API, lihat GetAccountSummarydi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getaccountsummary( ).
        MESSAGE 'Retrieved account summary.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting account summary.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetAccountSummarydi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetContextKeysForCustomPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Contoh 1: Untuk mencantumkan kunci konteks yang direferensikan oleh satu atau beberapa kebijakan JSON kustom yang disediakan sebagai parameter pada baris perintah

`get-context-keys-for-custom-policy`Perintah berikut mem-parsing setiap kebijakan yang disediakan dan mencantumkan kunci konteks yang digunakan oleh kebijakan tersebut. Gunakan perintah ini untuk mengidentifikasi nilai kunci konteks mana yang harus Anda berikan agar berhasil menggunakan perintah simulator kebijakan `simulate-custom-policy` dan`simulate-custom-policy`. Anda juga dapat mengambil daftar kunci konteks yang digunakan oleh semua kebijakan yang terkait oleh pengguna IAM atau peran dengan menggunakan perintah. `get-context-keys-for-custom-policy` Nilai parameter yang dimulai dengan `file://` menginstruksikan perintah untuk membaca file dan menggunakan konten sebagai nilai untuk parameter, bukan nama file itu sendiri.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Output:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Contoh 2: Untuk mencantumkan kunci konteks yang direferensikan oleh satu atau beberapa kebijakan JSON kustom yang disediakan sebagai input file

`get-context-keys-for-custom-policy`Perintah berikut ini sama dengan contoh sebelumnya, kecuali bahwa kebijakan disediakan dalam file, bukan sebagai parameter. Karena perintah mengharapkan daftar string JSON, dan bukan daftar struktur JSON, file harus terstruktur sebagai berikut, meskipun Anda dapat menciutkannya menjadi satu.

```
[
    "Policy1",
    "Policy2"
]
```

Jadi misalnya, file yang berisi kebijakan dari contoh sebelumnya harus terlihat seperti berikut. Anda harus menghindari setiap kutipan ganda yang disematkan di dalam string kebijakan dengan mendahuluinya dengan garis miring terbalik".

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

File ini kemudian dapat dikirimkan ke perintah berikut.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Output:

```
{
```

```
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan IAM Policy Simulator (AWS CLI AWS dan API) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetContextKeysForCustomPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam kebijakan yang disediakan JSON.Untuk memberikan beberapa kebijakan yang dapat Anda berikan sebagai daftar nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Untuk detail API, lihat GetContextKeysForCustomPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam kebijakan yang disediakan JSON.Untuk memberikan beberapa kebijakan yang dapat Anda berikan sebagai daftar nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

```
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Untuk detail API, lihat GetContextKeysForCustomPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetContextKeysForPrincipalPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Untuk membuat daftar kunci konteks yang direferensikan oleh semua kebijakan yang terkait dengan prinsipal IAM

`get-context-keys-for-principal-policy`Perintah berikut mengambil semua kebijakan yang dilampirkan ke pengguna `saanvi` dan grup mana pun yang menjadi anggotanya. Kemudian mem-parsing masing-masing dan mencantumkan kunci konteks yang digunakan oleh kebijakan tersebut. Gunakan perintah ini untuk mengidentifikasi nilai kunci konteks mana yang harus Anda berikan agar berhasil menggunakan `simulate-principal-policy` perintah `simulate-custom-policy` dan. Anda juga dapat mengambil daftar kunci konteks yang digunakan oleh kebijakan JSON arbitrer dengan menggunakan perintah. `get-context-keys-for-custom-policy`

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Output:

```
{
    "ContextKeyNames": [
        "aws:username",
```

```
        "aws:CurrentTime"
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan IAM Policy Simulator (AWS CLI AWS dan API) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetContextKeysForPrincipalPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam json kebijakan yang disediakan dan kebijakan yang dilampirkan ke entitas IAM (user/role dll.). Untuk -PolicyInputList Anda dapat memberikan beberapa daftar nilai sebagai nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Untuk detail API, lihat GetContextKeysForPrincipalPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil semua kunci konteks yang ada dalam json kebijakan yang disediakan dan kebijakan yang dilampirkan ke entitas IAM (user/role dll.). Untuk -PolicyInputList Anda dapat memberikan beberapa daftar nilai sebagai nilai yang dipisahkan koma.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

```
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Untuk detail API, lihat GetContextKeysForPrincipalPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetCredentialReport`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetCredentialReport`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

CLI

AWS CLI

Untuk mendapatkan laporan kredensi

Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks.

```
aws iam get-credential-report
```

Output:

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

Untuk informasi selengkapnya, lihat Mendapatkan laporan kredensi untuk AWS akun Anda di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetCredentialReportdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks. Baris pertama adalah header dengan nama kolom yang dipisahkan koma. Setiap baris berturut-turut adalah baris detail untuk satu pengguna, dengan setiap bidang dipisahkan dengan koma. Sebelum Anda dapat melihat laporan, Anda harus membuatnya dengan **`Request-IAMCredentialReport`** cmdlet. Untuk mengambil laporan sebagai string tunggal, gunakan **`-Raw`** sebagai pengganti. **`-AsTextArray`** Alias juga **`-SplitLines`** diterima untuk **`-AsTextArray`** sakelar. Untuk daftar lengkap kolom dalam output, lihat referensi API layanan. Perhatikan bahwa jika Anda tidak menggunakan **`-AsTextArray`** atau**`-SplitLines`**, maka Anda harus mengekstrak teks dari **`.Content`** properti menggunakan **`StreamReader`** kelas.NET.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Output:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
```

```
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Untuk detail API, lihat GetCredentialReportdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuka laporan yang dikembalikan dan mengeluarkannya ke pipeline sebagai array baris teks. Baris pertama adalah header dengan nama kolom yang dipisahkan koma. Setiap baris berturut-turut adalah baris detail untuk satu pengguna, dengan setiap bidang dipisahkan dengan koma. Sebelum Anda dapat melihat laporan, Anda harus membuatnya dengan **`Request-IAMCredentialReport`** cmdlet. Untuk mengambil laporan sebagai string tunggal, gunakan **`-Raw`** sebagai pengganti. **`-AsTextArray`** Alias juga **`-SplitLines`** diterima untuk **`-AsTextArray`** sakelar. Untuk daftar lengkap kolom dalam output, lihat referensi API layanan. Perhatikan bahwa jika Anda tidak menggunakan **`-AsTextArray`** atau**`-SplitLines`**, maka Anda harus mengekstrak teks dari **`.Content`** properti menggunakan **`StreamReader`** kelas.NET.

```
Request-IAMCredentialReport
```

Output:

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Output:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
```

```
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Untuk detail API, lihat GetCredentialReportdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Untuk detail API, lihat GetCredentialReportdi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->getcredentialreport( ).
    MESSAGE 'Retrieved credential report.' TYPE 'I'.
  CATCH /aws1/cx_iamcredrptnotpresen00.
    MESSAGE 'Credential report not present.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptexpiredex.
    MESSAGE 'Credential report expired.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptnotreadyex.
    MESSAGE 'Credential report not ready.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when getting credential report.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat GetCredentialReportdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetGroup`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetGroup`.

CLI

AWS CLI

Untuk mendapatkan grup IAM

Contoh ini mengembalikan rincian tentang grup `Admins` IAM.

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

Output:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

Untuk informasi selengkapnya, lihat Identitas IAM (pengguna, grup pengguna, dan peran) di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan detail tentang grup IAM**Testers**, termasuk kumpulan semua pengguna IAM yang termasuk dalam grup.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Untuk detail API, lihat GetGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan detail tentang grup IAM**`Testers`**, termasuk kumpulan semua pengguna IAM yang termasuk dalam grup.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Output:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Untuk detail API, lihat GetGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetGroupPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetGroupPolicy`.

CLI

AWS CLI

Untuk mendapatkan informasi tentang kebijakan yang dilampirkan ke grup IAM

`get-group-policy`Perintah berikut mendapatkan informasi tentang kebijakan tertentu yang dilampirkan ke grup bernama`Test-Group`.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Output:

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Untuk informasi selengkapnya, lihat Mengelola kebijakan IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan inline tertanam yang dinamai **`PowerUserAccess-Testers`** untuk grup**`Testers`**. **`PolicyDocument`**Properti ini dikodekan URL. Ini diterjemahkan dalam contoh ini dengan **`UrlDecode`** metode.NET.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Output:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Untuk detail API, lihat GetGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan inline tertanam yang dinamai **`PowerUserAccess-Testers`** untuk grup**`Testers`**. **`PolicyDocument`**Properti ini dikodekan URL. Ini diterjemahkan dalam contoh ini dengan **`UrlDecode`** metode.NET.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Output:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Untuk detail API, lihat GetGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetInstanceProfile`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan streaming terkelola

CLI

AWS CLI

Untuk mendapatkan informasi tentang profil instance

`get-instance-profile`Perintah berikut mendapatkan informasi tentang profil instance bernama`ExampleInstanceProfile`.

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Output:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
```

```
    }
}
```

Untuk informasi lebih lanjut, lihat Menggunakan profil instance dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian profil contoh bernama **ec2instancerole** yang didefinisikan dalam AWS akun saat ini.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk detail API, lihat GetInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian profil contoh bernama **ec2instancerole** yang didefinisikan dalam AWS akun saat ini.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
```

```
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk detail API, lihat GetInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetLoginProfile`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetLoginProfile`.

CLI

AWS CLI

Untuk mendapatkan informasi kata sandi untuk pengguna IAM

`get-login-profile`Perintah berikut mendapatkan informasi tentang password untuk pengguna IAM bernamaBob.

```
aws iam get-login-profile \
    --user-name Bob
```

Output:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

`get-login-profile`Perintah ini dapat digunakan untuk memverifikasi bahwa pengguna IAM memiliki kata sandi. Perintah mengembalikan `NoSuchEntity` kesalahan jika tidak ada kata sandi yang ditentukan untuk pengguna.

Anda tidak dapat melihat kata sandi menggunakan perintah ini. Jika kata sandi hilang, Anda dapat mengatur ulang kata sandi (`update-login-profile`) untuk pengguna. Atau, Anda dapat menghapus profil login (`delete-login-profile`) untuk pengguna dan kemudian membuat yang baru (`create-login-profile`).

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan tanggal pembuatan kata sandi dan apakah reset kata sandi diperlukan untuk pengguna **`David`** IAM.

```
Get-IAMLoginProfile -UserName David
```

Output:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- Untuk detail API, lihat GetLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan tanggal pembuatan kata sandi dan apakah reset kata sandi diperlukan untuk pengguna **`David`** IAM.

```
Get-IAMLoginProfile -UserName David
```

Output:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- Untuk detail API, lihat GetLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetOpenIdConnectProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetOpenIdConnectProvider`.

CLI

AWS CLI

Untuk mengembalikan informasi tentang penyedia OpenID Connect yang ditentukan

Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yang ARN. `arn:aws:iam::123456789012:oidc-provider/server.example.com`

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Output:

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat GetOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yang ARN adalah. **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`** **`ClientIDList`**Properti adalah koleksi yang berisi semua ID Klien yang ditentukan untuk penyedia ini.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Output:

```
ClientIDList         CreateDate                ThumbprintList
           Url
------------         ----------                --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Untuk detail API, lihat GetOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian tentang penyedia OpenID Connect yang ARN adalah. **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`** **`ClientIDList`**Properti adalah koleksi yang berisi semua ID Klien yang ditentukan untuk penyedia ini.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Output:

```
ClientIDList         CreateDate                ThumbprintList
           Url
------------         ----------                --------------
           ---
```

```
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Untuk detail API, lihat GetOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan streaming terkelola
- Bekerja dengan API Pembuat Kebijakan IAM

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {
```

```
        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
{ PolicyArn = policyArn });
        return response.Policy;
    }
```

- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengambil informasi tentang kebijakan terkelola yang ditentukan

Contoh ini mengembalikan rincian tentang kebijakan terkelola yang ARN. `arn:aws:iam::123456789012:policy/MySamplePolicy`

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetPolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
```

```
 }
 return policy, err
}
```

- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan kebijakan.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Untuk detail API, lihat GetPolicydi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
```

```
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan terkelola yang ARN. **`arn:aws:iam::123456789012:policy/MySamplePolicy`**

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Untuk detail API, lihat GetPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian tentang kebijakan terkelola yang ARN. **`arn:aws:iam::123456789012:policy/MySamplePolicy`**

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Untuk detail API, lihat GetPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
```

```
    else:
        return policy_statement
```

- Untuk detail API, lihat GetPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end
```

- Untuk detail API, lihat GetPolicydi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getpolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetPolicydi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
```

```
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: getPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi GetPolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetPolicyVersion`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetPolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Bekerja dengan API Pembuat Kebijakan IAM

CLI

AWS CLI

Untuk mengambil informasi tentang versi tertentu dari kebijakan terkelola yang ditentukan

Contoh ini mengembalikan dokumen kebijakan untuk versi v2 dari kebijakan yang ARN adalah. `arn:aws:iam::123456789012:policy/MyManagedPolicy`

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Output:

```
{
    "PolicyVersion": {
        "Document": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetPolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan dokumen kebijakan untuk **v2** versi kebijakan yang ARN-nya. **arn:aws:iam::123456789012:policy/MyManagedPolicy** Dokumen kebijakan dalam **Document** properti adalah URL yang dikodekan dan diterjemahkan dalam contoh ini dengan metode.NET. **UrlDecode**

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Output:

```
CreateDate             Document
 IsDefaultVersion     VersionId
```

```
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- Untuk detail API, lihat GetPolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan dokumen kebijakan untuk **v2** versi kebijakan yang ARN-nya. **arn:aws:iam::123456789012:policy/MyManagedPolicy** Dokumen kebijakan dalam **Document** properti adalah URL yang dikodekan dan diterjemahkan dalam contoh ini dengan metode.NET. **UrlDecode**

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Output:

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
```

```
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
          v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- Untuk detail API, lihat GetPolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
```

```
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
policy_arn)
        raise
    else:
        return policy_statement
```

- Untuk detail API, lihat GetPolicyVersiondi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getpolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Retrieved policy version information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iaminvalidinputex.
        MESSAGE 'Invalid input provided.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetPolicyVersiondi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Konfigurasikan konektivitas layanan kontainer
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Membuat layanan kontainer untuk instance mesin virtual
- Memulai dengan gudang data tanpa server
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
/// <summary>
/// Get information about an IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role to retrieve information
/// for.</param>
```

```
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }
```

- Untuk detail API, lihat GetRoledi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk mendapatkan informasi tentang peran IAM

`get-role`Perintah berikut mendapatkan informasi tentang peran bernama`Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Output:

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
```

```
    }
}
```

Perintah menampilkan kebijakan kepercayaan yang dilampirkan pada peran. Untuk mencantumkan kebijakan izin yang dilampirkan ke peran, gunakan `list-role-policies` perintah.

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetRoledi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
```

```
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Untuk detail API, lihat GetRoledi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan peran.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });
```

```
  return client.send(command);
};
```

- Untuk detail API, lihat GetRoledi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Untuk detail API, lihat GetRoledi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian**`lamda_exec_role`**. Ini termasuk dokumen kebijakan kepercayaan yang menentukan siapa yang dapat mengambil peran ini. Dokumen kebijakan adalah URL yang dikodekan dan dapat diterjemahkan menggunakan metode.NET. **`UrlDecode`** Dalam contoh ini, kebijakan asli menghapus semua spasi putih sebelum diunggah ke kebijakan. Untuk melihat dokumen kebijakan izin yang menentukan apa yang dapat dilakukan oleh seseorang yang mengasumsikan peran tersebut, gunakan kebijakan

`Get-IAMRolePolicy` for inline, dan **`Get-IAMPolicyVersion`** untuk kebijakan terkelola terlampir.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Output:

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Output:

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Untuk detail API, lihat GetRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian**`lamda_exec_role`**. Ini termasuk dokumen kebijakan kepercayaan yang menentukan siapa yang dapat mengambil peran ini. Dokumen kebijakan adalah URL yang dikodekan dan dapat diterjemahkan menggunakan metode.NET. **`UrlDecode`** Dalam contoh ini, kebijakan asli menghapus semua spasi putih sebelum diunggah ke kebijakan. Untuk melihat dokumen kebijakan izin yang menentukan apa yang dapat dilakukan oleh seseorang yang mengasumsikan peran tersebut, gunakan kebijakan

`Get-IAMRolePolicy` for inline, dan **`Get-IAMPolicyVersion`** untuk kebijakan terkelola terlampir.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Output:

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Output:

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Untuk detail API, lihat GetRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Untuk detail API, lihat GetRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Gets data about a role.
  #
  # @param name [String] The name of the role to look up.
  # @return [Aws::IAM::Role] The retrieved role.
  def get_role(name)
    role = @iam_client.get_role({
                                  role_name: name
                                }).role
    puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't get data for role '#{name}' Here's why:")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end
```

- Untuk detail API, lihat GetRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Untuk detail API, lihat GetRolereferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->getrole( iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved role information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetRoledi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getRole(name: String) async throws -> IAMClientTypes.Role {
        let input = GetRoleInput(
            roleName: name
        )
        do {
            let output = try await client.getRole(input: input)
```

```
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: getRole:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi GetRole AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetRolePolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetRolePolicy`.

CLI

AWS CLI

Untuk mendapatkan informasi tentang kebijakan yang dilampirkan pada peran IAM

`get-role-policy`Perintah berikut mendapatkan informasi tentang kebijakan tertentu yang dilampirkan pada peran bernama`Test-Role`.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Output:

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
```

```
                "s3:ListBucket",
                "s3:Put*",
                "s3:Get*",
                "s3:*MultipartUpload*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "1"
        }
    ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetRolePolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan dokumen kebijakan izin untuk kebijakan bernama **`oneClick_lambda_exec_role_policy`** yang disematkan dalam peran IAM. **`lamda_exec_role`** Dokumen kebijakan yang dihasilkan adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **`UrlDecode`** metode.NET.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Output:

```
PolicyDocument                                          PolicyName
              UserName
--------------                                          ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
```

```
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Output:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Untuk detail API, lihat GetRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan dokumen kebijakan izin untuk kebijakan bernama **`oneClick_lambda_exec_role_policy`** yang disematkan dalam peran IAM. **`lamda_exec_role`** Dokumen kebijakan yang dihasilkan adalah URL yang dikodekan. Ini diterjemahkan dalam contoh ini dengan **`UrlDecode`** metode.NET.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Output:

```
PolicyDocument                                                  PolicyName
              UserName
--------------                                                  ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Output:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Untuk detail API, lihat GetRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetSamlProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetSamlProvider`.

CLI

AWS CLI

Untuk mengambil metadocument penyedia SAFL

Contoh ini mengambil rincian tentang penyedia SAFL 2.0 yang ARM. `arn:aws:iam::123456789012:saml-provider/SAMLADFS` Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas penyedia AWS SAFL serta tanggal pembuatan dan kedaluwarsa.

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas IAM SALL di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetSamlProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil detail tentang penyedia SAFL 2.0 yang ARM adalah arn:aws:iam: :123456789012: saml-. provider/SAMLADFS Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas penyedia AWS SAFL serta tanggal pembuatan dan kedaluwarsa.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Output:

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Untuk detail API, lihat GetSamlProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil detail tentang penyedia SAFL 2.0 yang ARM adalah arn:aws:iam: :123456789012: saml-. provider/SAMLADFS Responsnya mencakup dokumen metadata yang Anda dapatkan dari penyedia identitas untuk membuat entitas penyedia AWS SAFL serta tanggal pembuatan dan kedaluwarsa.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Output:

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
```

```
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Untuk detail API, lihat GetSamlProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetServerCertificate`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServerCertificate`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

```
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }

    return result;
}
```

- Untuk detail API, lihat GetServerCertificatedi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mendapatkan detail tentang sertifikat server di AWS akun Anda

`get-server-certificate`Perintah berikut mengambil semua detail tentang sertifikat server yang ditentukan di AWS akun Anda.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Output:

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
```

```
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```


Untuk membuat daftar sertifikat server yang tersedia di AWS akun Anda, gunakan `list-server-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server di IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan sertifikat server.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetServerCertificatedi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat GetServerCertificatedi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang sertifikat server bernama**`MyServerCertificate`**. Anda dapat menemukan rincian sertifikat di **`CertificateBody`** dan **`ServerCertificateMetadata`** properti.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Output:

```
CertificateBody          : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                           21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                           rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                           nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                           NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                           -----END CERTIFICATE-----
CertificateChain         :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat GetServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil rincian tentang sertifikat server bernama**`MyServerCertificate`**. Anda dapat menemukan rincian sertifikat di **`CertificateBody`** dan **`ServerCertificateMetadata`** properti.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Output:

```
CertificateBody          : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat GetServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetServiceLastAccessedDetails`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLastAccessedDetails`.

CLI

AWS CLI

Untuk mengambil laporan akses layanan

`get-service-last-accessed-details`Contoh berikut mengambil laporan yang dihasilkan sebelumnya yang mencantumkan layanan yang diakses oleh entitas IAM. Untuk menghasilkan laporan, gunakan `generate-service-last-accessed-details` perintah.

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Output:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetServiceLastAccessedDetailsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan rincian layanan yang terakhir diakses oleh entitas IAM (pengguna, grup, peran, atau kebijakan) yang terkait dalam panggilan Permintaan.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Output:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Untuk detail API, lihat GetServiceLastAccessedDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memberikan rincian layanan yang terakhir diakses oleh entitas IAM (pengguna, grup, peran, atau kebijakan) yang terkait dalam panggilan Permintaan.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Output:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Untuk detail API, lihat GetServiceLastAccessedDetailsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetServiceLastAccessedDetailsWithEntities`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Untuk mengambil laporan akses layanan dengan rincian untuk layanan

`get-service-last-accessed-details-with-entities`Contoh berikut mengambil laporan yang berisi rincian tentang pengguna IAM dan entitas lain yang mengakses layanan yang ditentukan. Untuk menghasilkan laporan, gunakan `generate-service-last-`

`accessed-details` perintah. Untuk mendapatkan daftar layanan yang diakses dengan ruang nama, gunakan. `get-service-last-accessed-details`

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Output:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menyempurnakan izin dalam AWS menggunakan informasi yang terakhir diakses di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetServiceLastAccessedDetailsWithEntitiesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memberikan stempel waktu terakhir yang diakses untuk layanan dalam permintaan oleh entitas IAM masing-masing.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Output:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Output:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Output:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Untuk detail API, lihat GetServiceLastAccessedDetailsWithEntitiesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memberikan stempel waktu terakhir yang diakses untuk layanan dalam permintaan oleh entitas IAM masing-masing.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Output:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Output:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Output:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Untuk detail API, lihat GetServiceLastAccessedDetailsWithEntitiesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetServiceLinkedRoleDeletionStatus`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

Untuk memeriksa status permintaan untuk menghapus peran terkait layanan

`get-service-linked-role-deletion-status`Contoh berikut menampilkan status permintaan sebelumnya untuk menghapus peran terkait layanan. Operasi hapus terjadi secara asinkron. Ketika Anda membuat permintaan, Anda mendapatkan `DeletionTaskId` nilai yang Anda berikan sebagai parameter untuk perintah ini.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Output:

```
{
"Status": "SUCCEEDED"
}
```

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetServiceLinkedRoleDeletionStatusdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetServiceLinkedRoleDeletionStatusdi Referensi AWS SDK untuk JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetUser`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetUser`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Untuk detail API, lihat GetUserdi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Untuk detail API, lihat GetUserdi Referensi AWS CLI Perintah.

CLI

AWS CLI

Untuk mendapatkan informasi tentang pengguna IAM

`get-user`Perintah berikut mendapatkan informasi tentang nama `Paulo` pengguna IAM.

```
aws iam get-user \
    --user-name Paulo
```

Output:

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Untuk informasi selengkapnya, lihat Mengelola pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat GetUserdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Untuk detail API, lihat GetUserdi Referensi AWS SDK untuk Go API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang nama **David** pengguna.

```
Get-IAMUser -UserName David
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Contoh 2: Contoh ini mengambil detail tentang pengguna IAM yang saat ini masuk.

```
Get-IAMUser
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Untuk detail API, lihat GetUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil rincian tentang nama **David** pengguna.

```
Get-IAMUser -UserName David
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Contoh 2: Contoh ini mengambil detail tentang pengguna IAM yang saat ini masuk.

```
Get-IAMUser
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Untuk detail API, lihat GetUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Retrieves a user's details
  #
  # @param user_name [String] The name of the user to retrieve
  # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
  def get_user(user_name)
```

```
    response = @iam_client.get_user(user_name: user_name)
    response.user
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("User '#{user_name}' not found.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error retrieving user '#{user_name}': #{e.message}")
    nil
  end
```

- Untuk detail API, lihat GetUserdi Referensi AWS SDK untuk Ruby API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func getUser(name: String? = nil) async throws -> IAMClientTypes.User
 {
        let input = GetUserInput(
            userName: name
        )
        do {
            let output = try await iamClient.getUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            return user
        } catch {
            print("ERROR: getUser:", dump(error))
            throw error
        }
```

```
    }
```

- Untuk detail API, lihat referensi GetUser AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetUserPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetUserPolicy`.

CLI

AWS CLI

Untuk mencantumkan detail kebijakan untuk pengguna IAM

`get-user-policy`Perintah berikut mencantumkan rincian kebijakan tertentu yang dilampirkan ke pengguna IAM bernamaBob.

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Output:

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Untuk mendapatkan daftar kebijakan untuk pengguna IAM, gunakan `list-user-policies` perintah.

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetUserPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian kebijakan inline bernama **`Davids_IAM_Admin_Policy`** yang disematkan dalam nama pengguna IAM. **`David`** Dokumen kebijakan adalah URL yang dikodekan.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Output:

```
PolicyDocument                                          PolicyName
       UserName
--------------                                          ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
```

```
    }
  ]
}
```

- Untuk detail API, lihat GetUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil rincian kebijakan inline bernama **Davids_IAM_Admin_Policy** yang disematkan dalam nama pengguna IAM. **David** Dokumen kebijakan adalah URL yang dikodekan.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Output:

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- Untuk detail API, lihat GetUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListAccessKeys`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAccessKeys`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
```

```
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
```

```
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
```

```
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mencantumkan ID kunci akses untuk pengguna IAM

`list-access-keys`Perintah berikut mencantumkan ID kunci akses untuk pengguna IAM bernama`Bob`.

```
aws iam list-access-keys \
    --user-name Bob
```

Output:

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Anda tidak dapat mencantumkan kunci akses rahasia untuk pengguna IAM. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru menggunakan `create-access-keys` perintah.

Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
```

```
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s
```

```
                Where:
                    userName - The name of the user for which access keys are
retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();
```

```
                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
 metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kunci akses.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Untuk detail API, lihat ListAccessKeysdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mencantumkan kunci akses untuk pengguna IAM bernama**Bob**. Perhatikan bahwa Anda tidak dapat mencantumkan kunci akses rahasia untuk pengguna IAM. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru dengan **New-IAMAccessKey** cmdlet.

```
Get-IAMAccessKey -UserName "Bob"
```

Output:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Untuk detail API, lihat ListAccessKeysdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mencantumkan kunci akses untuk pengguna IAM bernama**Bob**. Perhatikan bahwa Anda tidak dapat mencantumkan kunci akses rahasia untuk pengguna IAM. Jika kunci akses rahasia hilang, Anda harus membuat kunci akses baru dengan **`New-IAMAccessKey`** cmdlet.

```
Get-IAMAccessKey -UserName "Bob"
```

Output:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Untuk detail API, lihat ListAccessKeysdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_keys(user_name):
```

```
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Untuk detail API, lihat ListAccessKeysdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
```

```
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
```

```
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat ListAccessKeysdi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listaccesskeys(
          iv_username = iv_user_name ).
        MESSAGE 'Retrieved access key list.' TYPE 'I'.
```

```
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListAccessKeysdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListAccountAliases`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAccountAliases`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola akun Anda

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk membuat daftar alias akun

`list-account-aliases`Perintah berikut mencantumkan alias untuk akun saat ini.

```
aws iam list-account-aliases
```

Output:

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Untuk informasi selengkapnya, lihat ID AWS akun Anda dan aliasnya di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
```

```
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar alias akun.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
```

```
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listAliases() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Untuk detail API, lihat ListAccountAliasesdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan alias akun untuk. Akun AWS

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Untuk detail API, lihat ListAccountAliasesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mengembalikan alias akun untuk. Akun AWS

```
Get-IAMAccountAlias
```

Output:

```
ExampleCo
```

- Untuk detail API, lihat ListAccountAliasesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_aliases():
    """
```

```
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- Untuk detail API, lihat ListAccountAliasesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
```

```
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat ListAccountAliasesdi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->listaccountaliases( ).
    MESSAGE 'Retrieved account alias list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing account aliases.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat ListAccountAliasesdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListAttachedGroupPolicies`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedGroupPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke grup yang ditentukan

Contoh ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan ke grup IAM yang disebutkan `Admins` di AWS akun.

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Output:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListAttachedGroupPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan ke grup IAM yang disebutkan **`Admins`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam grup, gunakan **`Get-IAMGroupPolicyList`** perintah.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Untuk detail API, lihat ListAttachedGroupPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan ke grup IAM yang disebutkan **`Admins`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam grup, gunakan **`Get-IAMGroupPolicyList`** perintah.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Untuk detail API, lihat ListAttachedGroupPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListAttachedRolePolicies`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedRolePolicies`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
```

```
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan pada peran yang ditentukan

Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan pada peran IAM yang disebutkan `SecurityAuditRole` di AWS akun.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Output:

```
{
    "AttachedPolicies": [
        {
```

```
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}
```

```
// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan yang dilampirkan pada peran.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan pada peran IAM yang disebutkan **`SecurityAuditRole`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam peran, gunakan **`Get-IAMRolePolicyList`** perintah.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola yang dilampirkan pada peran IAM yang disebutkan **`SecurityAuditRole`** di AWS akun. Untuk melihat daftar kebijakan sebaris yang disematkan dalam peran, gunakan **`Get-IAMRolePolicyList`** perintah.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
```

```
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
```

```
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
```

```
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_attached_role_policies(
```

```
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk detail API, lihat ListAttachedRolePoliciesreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listattachedrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved attached policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListAttachedRolePoliciesdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3



    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
 [IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []

        // Use "Paginated" to get all the attached role polices.
        // This lets the SDK handle the 'isTruncated' in
 "ListAttachedRolePoliciesOutput".
        let input = ListAttachedRolePoliciesInput(
            roleName: role
        )
        let output = client.listAttachedRolePoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let attachedPolicies = page.attachedPolicies else {
                    print("Error: no attached policies returned.")
                    continue
                }
```

```
                for attachedPolicy in attachedPolicies {
                    policyList.append(attachedPolicy)
                }
            }
        } catch {
            print("ERROR: listAttachedRolePolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Untuk detail API, lihat referensi ListAttachedRolePolicies AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListAttachedUserPolicies`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListAttachedUserPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan terkelola yang dilampirkan ke pengguna yang ditentukan

Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola untuk pengguna IAM yang disebutkan Bob di AWS akun.

```
aws iam list-attached-user-policies \
    --user-name Bob
```

Output:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
```

```
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListAttachedUserPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola untuk pengguna IAM yang disebutkan **Bob** di AWS akun. Untuk melihat daftar kebijakan inline yang disematkan di pengguna IAM, gunakan perintah. **Get-IAMUserPolicyList**

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Output:

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/TesterPolicy                     TesterPolicy
```

- Untuk detail API, lihat ListAttachedUserPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Perintah ini mengembalikan nama dan ARN dari kebijakan terkelola untuk pengguna IAM yang disebutkan **Bob** di AWS akun. Untuk melihat daftar kebijakan inline yang disematkan di pengguna IAM, gunakan perintah. **Get-IAMUserPolicyList**

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Output:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/TesterPolicy                      TesterPolicy
```

- Untuk detail API, lihat ListAttachedUserPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListEntitiesForPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListEntitiesForPolicy`.

CLI

AWS CLI

Untuk mencantumkan semua pengguna, grup, dan peran yang dilampirkan oleh kebijakan terkelola yang ditentukan

Contoh ini menampilkan daftar grup, peran, dan pengguna IAM yang memiliki kebijakan `arn:aws:iam::123456789012:policy/TestPolicy` terlampir.

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Output:

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
```

```
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListEntitiesForPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menampilkan daftar grup, peran, dan pengguna IAM yang memiliki kebijakan yang **`arn:aws:iam::123456789012:policy/TestPolicy`** dilampirkan.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Output:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Untuk detail API, lihat ListEntitiesForPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menampilkan daftar grup, peran, dan pengguna IAM yang memiliki kebijakan yang **`arn:aws:iam::123456789012:policy/TestPolicy`** dilampirkan.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Output:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Untuk detail API, lihat ListEntitiesForPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListGroupPolicies`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroupPolicies`.

CLI

AWS CLI

Untuk mencantumkan semua kebijakan inline yang dilampirkan ke grup yang ditentukan

`list-group-policies`Perintah berikut mencantumkan nama-nama kebijakan inline yang dilampirkan ke grup IAM yang disebutkan `Admins` di akun saat ini.

```
aws iam list-group-policies \
    --group-name Admins
```

Output:

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
```

```
}
```

Untuk informasi selengkapnya, lihat Mengelola kebijakan IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListGroupPoliciesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar kebijakan inline yang disematkan dalam grup**`Testers`**. Untuk mendapatkan kebijakan terkelola yang dilampirkan ke grup, gunakan perintah**`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Output:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Untuk detail API, lihat ListGroupPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan daftar kebijakan inline yang disematkan dalam grup**`Testers`**. Untuk mendapatkan kebijakan terkelola yang dilampirkan ke grup, gunakan perintah**`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Output:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Untuk detail API, lihat ListGroupPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListGroups`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroups`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Untuk detail API, lihat ListGroupsdi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk membuat daftar grup IAM untuk akun saat ini

`list-groups`Perintah berikut mencantumkan grup IAM di akun saat ini.

```
aws iam list-groups
```

Output:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListGroupsdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(ctx context.Context, maxGroups int32)
 ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(ctx, &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
```

```
 return groups, err
}
```

- Untuk detail API, lihat ListGroupsdi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar grup.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
```

```
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk detail API, lihat ListGroupsdi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
```

```
    }
```

- Untuk detail API, lihat ListGroupsdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan koleksi semua grup IAM didefinisikan dalam saat ini Akun AWS.

```
Get-IAMGroupList
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Untuk detail API, lihat ListGroupsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan koleksi semua grup IAM didefinisikan dalam saat ini Akun AWS.

```
Get-IAMGroupList
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Untuk detail API, lihat ListGroupsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
```

```
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Untuk detail API, lihat ListGroupsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

```
end
```

- Untuk detail API, lihat ListGroupsdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk detail API, lihat ListGroupsreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listgroups( ).
        MESSAGE 'Retrieved group list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing groups.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListGroupsdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listGroups() async throws -> [String] {
        var groupList: [String] = []

        // Use "Paginated" to get all the groups.
        // This lets the SDK handle the 'isTruncated' property in
 "ListGroupsOutput".
```

```
        let input = ListGroupsInput()

        let pages = client.listGroupsPaginated(input: input)
        do {
            for try await page in pages {
                guard let groups = page.groups else {
                    print("Error: no groups returned.")
                    continue
                }

                for group in groups {
                    if let name = group.groupName {
                        groupList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listGroups:", dump(error))
            throw error
        }
        return groupList
    }
```

- Untuk detail API, lihat referensi ListGroups AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListGroupsForUser`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListGroupsForUser`.

CLI

AWS CLI

Untuk membuat daftar grup yang dimiliki pengguna IAM

`list-groups-for-user`Perintah berikut menampilkan kelompok-kelompok yang bernama pengguna IAM Bob milik.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Output:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListGroupsForUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar grup IAM yang **David** dimiliki pengguna IAM.

```
Get-IAMGroupForUser -UserName David
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
```

```
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Untuk detail API, lihat ListGroupsForUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan daftar grup IAM yang **`David`** dimiliki pengguna IAM.

```
Get-IAMGroupForUser -UserName David
```

Output:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Untuk detail API, lihat ListGroupsForUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListInstanceProfiles`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListInstanceProfiles`.

CLI

AWS CLI

Untuk mencantumkan profil instance untuk akun

`list-instance-profiles`Perintah berikut mencantumkan profil instance yang terkait dengan akun saat ini.

```
aws iam list-instance-profiles
```

Output:

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
```

```
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

Untuk informasi lebih lanjut, lihat Menggunakan profil instance dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListInstanceProfilesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan koleksi profil instance didefinisikan dalam saat ini Akun AWS.

```
Get-IAMInstanceProfileList
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk detail API, lihat ListInstanceProfilesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan koleksi profil instance didefinisikan dalam saat ini Akun AWS.

```
Get-IAMInstanceProfileList
```

Output:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Untuk detail API, lihat ListInstanceProfilesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListInstanceProfilesForRole`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListInstanceProfilesForRole`.

CLI

AWS CLI

Untuk membuat daftar profil instance untuk peran IAM

`list-instance-profiles-for-role`Perintah berikut mencantumkan profil instance yang terkait dengan peran tersebut`Test-Role`.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Output:

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
```

```
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Untuk informasi lebih lanjut, lihat Menggunakan profil instance dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListInstanceProfilesForRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian profil instance yang terkait dengan peran**ec2instancerole**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Output:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Untuk detail API, lihat ListInstanceProfilesForRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian profil instance yang terkait dengan peran**ec2instancerole**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Output:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Untuk detail API, lihat ListInstanceProfilesForRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListMfaDevices`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListMfaDevices`.

CLI

AWS CLI

Untuk mencantumkan semua perangkat MFA untuk pengguna tertentu

Contoh ini mengembalikan rincian tentang perangkat MFA yang ditetapkan untuk pengguna IAM. Bob

```
aws iam list-mfa-devices \
    --user-name Bob
```

Output:

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
```

```
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multifaktor (MFA) dalam AWS dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListMfaDevicesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan rincian tentang perangkat MFA yang ditetapkan untuk pengguna IAM. **`David`** Dalam contoh ini Anda dapat mengatakan bahwa itu adalah perangkat virtual karena **`SerialNumber`** adalah ARN bukan nomor seri aktual perangkat fisik.

```
Get-IAMMFADevice -UserName David
```

Output:

```
EnableDate                SerialNumber                           UserName
----------                ------------                           --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David    David
```

- Untuk detail API, lihat ListMfaDevicesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan rincian tentang perangkat MFA yang ditetapkan untuk pengguna IAM. **`David`** Dalam contoh ini Anda dapat mengatakan bahwa itu adalah perangkat virtual karena **`SerialNumber`** adalah ARN bukan nomor seri aktual perangkat fisik.

```
Get-IAMMFADevice -UserName David
```

Output:

```
EnableDate                  SerialNumber                            UserName
----------                  ------------                            --------
4/8/2015 9:41:10 AM         arn:aws:iam::123456789012:mfa/David     David
```

- Untuk detail API, lihat ListMfaDevicesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListOpenIdConnectProviders`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListOpenIdConnectProviders`.

CLI

AWS CLI

Untuk mencantumkan informasi tentang penyedia OpenID Connect di akun AWS

Contoh ini mengembalikan daftar ARNS dari semua penyedia OpenID Connect yang didefinisikan dalam akun AWS saat ini.

```
aws iam list-open-id-connect-providers
```

Output:

```
{
```

```
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat ListOpenIdConnectProvidersdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar ARNS dari semua penyedia OpenID Connect yang didefinisikan dalam Akun AWS saat ini.

```
Get-IAMOpenIDConnectProviderList
```

Output:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Untuk detail API, lihat ListOpenIdConnectProvidersdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan daftar ARNS dari semua penyedia OpenID Connect yang didefinisikan dalam Akun AWS saat ini.

```
Get-IAMOpenIDConnectProviderList
```

Output:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Untuk detail API, lihat ListOpenIdConnectProvidersdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListPolicies`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListPolicies`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
```

```
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
```

```
                        std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                        std::setw(64) << "Description" << std::setw(12) <<
                        "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mencantumkan kebijakan terkelola yang tersedia untuk AWS akun Anda

Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pertama yang tersedia di AWS akun saat ini.

```
aws iam list-policies \
    --max-items 3
```

Output:


```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListPoliciesdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
```

```
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
```

```
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
```

```
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan kumpulan dari tiga kebijakan terkelola pertama yang tersedia di AWS akun saat ini. Karena tidak **`-scope`** ditentukan, default ke **`all`** dan mencakup kebijakan terkelola dan yang AWS dikelola pelanggan.

```
Get-IAMPolicyList -MaxItem 3
```

Output:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
```

```
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Contoh 2: Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pelanggan pertama yang tersedia di AWS akun saat ini. Ini digunakan **`-Scope local`** untuk membatasi output hanya pada kebijakan yang dikelola pelanggan.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
```

```
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Untuk detail API, lihat ListPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan kumpulan dari tiga kebijakan terkelola pertama yang tersedia di AWS akun saat ini. Karena tidak **`-scope`** ditentukan, default ke **`all`** dan mencakup kebijakan terkelola dan yang AWS dikelola pelanggan.

```
Get-IAMPolicyList -MaxItem 3
```

Output:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM
```

```
Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Contoh 2: Contoh ini mengembalikan kumpulan dari dua kebijakan terkelola pelanggan pertama yang tersedia di AWS akun saat ini. Ini digunakan **`-Scope local`** untuk membatasi output hanya pada kebijakan yang dikelola pelanggan.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Output:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Untuk detail API, lihat ListPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Untuk detail API, lihat ListPoliciesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat ListPoliciesdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;
```

```
    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Untuk detail API, lihat ListPoliciesreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listpolicies( iv_scope = iv_scope ).
        MESSAGE 'Retrieved policy list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policies.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListPoliciesdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []

        // Use "Paginated" to get all the policies.
        // This lets the SDK handle the 'isTruncated' in "ListPoliciesOutput".
        let input = ListPoliciesInput()
        let output = client.listPoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let policies = page.policies else {
                    print("Error: no policies returned.")
                    continue
                }

                for policy in policies {
                    guard let name = policy.policyName,
                          let id = policy.policyId,
                          let arn = policy.arn
                    else {
                        throw ServiceHandlerError.noSuchPolicy
                    }
                    policyList.append(MyPolicyRecord(name: name, id: id, arn:
 arn))
                }
            }
        } catch {
            print("ERROR: listPolicies:", dump(error))
```

```
            throw error
        }

        return policyList
    }
```

- Untuk detail API, lihat referensi ListPolicies AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListPolicyVersions`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListPolicyVersions`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk mencantumkan informasi tentang versi kebijakan terkelola yang ditentukan

Contoh ini mengembalikan daftar versi kebijakan yang tersedia yang ARN-nya. `arn:aws:iam::123456789012:policy/MySamplePolicy`

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Output:

```
{
    "IsTruncated": false,
```

```
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ListPolicyVersionsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar versi kebijakan yang tersedia yang ARN-nya. **`arn:aws:iam::123456789012:policy/MyManagedPolicy`** Untuk mendapatkan dokumen kebijakan untuk versi tertentu, gunakan **`Get-IAMPolicyVersion`** perintah dan tentukan **`VersionId`** yang Anda inginkan.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Output:

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- Untuk detail API, lihat ListPolicyVersionsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan daftar versi kebijakan yang tersedia yang ARN-nya. **`arn:aws:iam::123456789012:policy/MyManagedPolicy`** Untuk mendapatkan dokumen kebijakan untuk versi tertentu, gunakan **`Get-IAMPolicyVersion`** perintah dan tentukan **`VersionId`** yang Anda inginkan.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Output:

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- Untuk detail API, lihat ListPolicyVersionsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
```

```
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListPolicyVersionsdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListRolePolicies`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRolePolicies`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan perlindungan perangkat internet of things

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
```

```
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
_IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk mencantumkan kebijakan yang dilampirkan pada peran IAM

`list-role-policies`Perintah berikut mencantumkan nama kebijakan izin untuk peran IAM yang ditentukan.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Output:

```
{
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

Untuk melihat kebijakan kepercayaan yang dilampirkan pada peran, gunakan `get-role` perintah. Untuk melihat detail kebijakan izin, gunakan `get-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar kebijakan.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });
```

```
  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
```

```
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengembalikan daftar nama kebijakan inline yang disematkan dalam peran IAM. **`lamda_exec_role`** Untuk melihat detail kebijakan inline, gunakan perintah**`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Output:

```
oneClick_lambda_exec_role_policy
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengembalikan daftar nama kebijakan inline yang disematkan dalam peran IAM. **`lamda_exec_role`** Untuk melihat detail kebijakan inline, gunakan perintah**`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Output:

```
oneClick_lambda_exec_role_policy
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Untuk detail API, lihat ListRolePoliciesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end
```

- Untuk detail API, lihat ListRolePoliciesdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Untuk detail API, lihat ListRolePoliciesreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    oo_result = lo_iam->listrolepolicies(
      iv_rolename = iv_role_name ).
    MESSAGE 'Retrieved inline policy list for role.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Role does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat ListRolePoliciesdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3
```

```
    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []

        // Use "Paginated" to get all the role policies.
        // This lets the SDK handle the 'isTruncated' in
"ListRolePoliciesOutput".
        let input = ListRolePoliciesInput(
            roleName: role
        )
        let pages = client.listRolePoliciesPaginated(input: input)

        do {
            for try await page in pages {
                guard let policies = page.policyNames else {
                    print("Error: no role policies returned.")
                    continue
                }

                for policy in policies {
                    policyList.append(policy)
                }
            }
        } catch {
            print("ERROR: listRolePolicies:", dump(error))
            throw error
        }
        return policyList
    }
```

- Untuk detail API, lihat referensi ListRolePolicies AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListRoleTags`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRoleTags`.

CLI

AWS CLI

Untuk mencantumkan tag yang dilampirkan pada peran

`list-role-tags`Perintah berikut mengambil daftar tag yang terkait dengan peran yang ditentukan.

```
aws iam list-role-tags \
    --role-name production-role
```

Output:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListRoleTagsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag yang terkait dengan peran..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Untuk detail API, lihat ListRoleTagsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil tag yang terkait dengan peran..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Untuk detail API, lihat ListRoleTagsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListRoles`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListRoles`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }
```

```
        return roles;
    }
```

- Untuk detail API, lihat ListRolesdi Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Untuk mencantumkan peran IAM untuk akun saat ini

`list-roles`Perintah berikut mencantumkan peran IAM untuk akun saat ini.

```
aws iam list-roles
```

Output:

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
```

```
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListRolesdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
```

```
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Untuk detail API, lihat ListRolesdi Referensi AWS SDK untuk Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar peran.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
```

```
      );
    } else {
      break;
    }
  }
}
```

- Untuk detail API, lihat ListRolesdi Referensi AWS SDK untuk JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
```

```
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Untuk detail API, lihat ListRolesdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar semua peran IAM di. Akun AWS

```
Get-IAMRoleList
```

- Untuk detail API, lihat ListRolesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil daftar semua peran IAM di. Akun AWS

```
Get-IAMRoleList
```

- Untuk detail API, lihat ListRolesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
```

```
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Untuk detail API, lihat ListRolesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count

      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
```

```
    end

    role_names
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat ListRolesdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Untuk detail API, lihat ListRolesreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listroles( ).
        MESSAGE 'Retrieved role list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing roles.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListRolesdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listRoles() async throws -> [String] {
        var roleList: [String] = []

        // Use "Paginated" to get all the roles.
        // This lets the SDK handle the 'isTruncated' in "ListRolesOutput".
        let input = ListRolesInput()
```

```
        let pages = client.listRolesPaginated(input: input)

        do {
            for try await page in pages {
                guard let roles = page.roles else {
                    print("Error: no roles returned.")
                    continue
                }

                for role in roles {
                    if let name = role.roleName {
                        roleList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listRoles:", dump(error))
            throw error
        }
        return roleList
    }
```

- Untuk detail API, lihat referensi ListRoles AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListSamlProviders`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListSAMLProviders`.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
 ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk .NET Referensi API.

CLI

AWS CLI

Untuk membuat daftar penyedia SAFL di akun AWS

Contoh ini mengambil daftar penyedia SAFL 2.0 yang dibuat di akun saat ini AWS .

```
aws iam list-saml-providers
```

Output:

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas IAM SALL di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListSamlProviders di AWS CLI Referensi Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk Go Referensi API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar penyedia SAFL.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk JavaScript Referensi API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk PHP Referensi API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar penyedia SAFL 2.0 yang dibuat saat ini. Akun AWS Ini mengembalikan ARN, tanggal pembuatan, dan tanggal kedaluwarsa untuk setiap penyedia SAFL.

```
Get-IAMSAMLProviderList
```

Output:

```
Arn                                                  CreateDate
   ValidUntil
---                                                  ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS     12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Untuk detail API, lihat ListSamlProviders di Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil daftar penyedia SAFL 2.0 yang dibuat saat ini. Akun AWS Ini mengembalikan ARN, tanggal pembuatan, dan tanggal kedaluwarsa untuk setiap penyedia SAFL.

```
Get-IAMSAMLProviderList
```

Output:

```
Arn                                                  CreateDate
   ValidUntil
---                                                  ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS     12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Untuk detail API, lihat ListSamlProviders di Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
```

```
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
```

```
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk Ruby Referensi API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Untuk detail API, lihat referensi ListSamlProviders di AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listsamlproviders( ).
        MESSAGE 'Retrieved SAML provider list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing SAML providers.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListSamlProviders di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListServerCertificates`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListServerCertificates`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk detail API, lihat ListServerCertificatesdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mencantumkan sertifikat server di AWS akun Anda

`list-server-certificates`Perintah berikut mencantumkan semua sertifikat server yang disimpan dan tersedia untuk digunakan di AWS akun Anda.

```
aws iam list-server-certificates
```

Output:

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola sertifikat server di IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListServerCertificatesdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat daftar sertifikat.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListServerCertificatesdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListServerCertificatesdi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar sertifikat server yang telah diunggah ke saat ini. Akun AWS

```
Get-IAMServerCertificateList
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat ListServerCertificatesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil daftar sertifikat server yang telah diunggah ke saat ini. Akun AWS

```
Get-IAMServerCertificateList
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat ListServerCertificatesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat ListServerCertificatesdi Referensi AWS SDK untuk Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListSigningCertificates`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListSigningCertificates`.

CLI

AWS CLI

Untuk mencantumkan sertifikat penandatanganan untuk pengguna IAM

`list-signing-certificates`Perintah berikut mencantumkan sertifikat penandatanganan untuk pengguna IAM bernama`Bob`.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Output:

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan Pengguna Amazon EC2.

- Untuk detail API, lihat ListSigningCertificatesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil rincian tentang sertifikat penandatanganan yang terkait dengan nama **`Bob`** pengguna.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk detail API, lihat ListSigningCertificatesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil rincian tentang sertifikat penandatanganan yang terkait dengan nama **Bob** pengguna.

```
Get-IAMSigningCertificate -UserName Bob
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk detail API, lihat ListSigningCertificatesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListUserPolicies`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUserPolicies`.

CLI

AWS CLI

Untuk membuat daftar kebijakan untuk pengguna IAM

`list-user-policies`Perintah berikut mencantumkan kebijakan yang dilampirkan ke nama Bob pengguna IAM.

```
aws iam list-user-policies \
    --user-name Bob
```

Output:

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Untuk informasi selengkapnya, lihat Membuat pengguna IAM di AWS akun Anda di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListUserPoliciesdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
```

```
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Untuk detail API, lihat ListUserPoliciesdi Referensi AWS SDK untuk Go API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil daftar nama kebijakan inline yang disematkan dalam nama pengguna IAM. **`David`**

```
Get-IAMUserPolicyList -UserName David
```

Output:

```
Davids_IAM_Admin_Policy
```

- Untuk detail API, lihat ListUserPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil daftar nama kebijakan inline yang disematkan dalam nama pengguna IAM. **`David`**

```
Get-IAMUserPolicyList -UserName David
```

Output:

```
Davids_IAM_Admin_Policy
```

- Untuk detail API, lihat ListUserPoliciesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListUserTags`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUserTags`.

CLI

AWS CLI

Untuk mencantumkan tag yang dilampirkan ke pengguna

`list-user-tags`Perintah berikut mengambil daftar tag yang terkait dengan pengguna IAM tertentu.

```
aws iam list-user-tags \
    --user-name alice
```

Output:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListUserTagsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil tag yang terkait dengan pengguna.

```
Get-IAMUserTagList -UserName joe
```

- Untuk detail API, lihat ListUserTagsdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil tag yang terkait dengan pengguna.

```
Get-IAMUserTagList -UserName joe
```

- Untuk detail API, lihat ListUserTagsdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListUsers`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListUsers`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
 ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
##############################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat ListUsersdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
```

```
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk daftar pengguna IAM

`list-users`Perintah berikut mencantumkan pengguna IAM di akun saat ini.

```
aws iam list-users
```

Output:

```
{
    "Users": [
```

```
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Daftar pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat ListUsersdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
```

```
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
```

```
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
 ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }
```

```
                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
 user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

 permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar pengguna.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);

  for (const { UserName, CreateDate } of response.Users) {
    console.log(`${UserName} created on: ${CreateDate}`);
  }
  return response;
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
```

```
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun listAllUsers() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- Untuk detail API, lihat ListUsersdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk PHP API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil koleksi pengguna saat ini Akun AWS.

```
Get-IAMUserList
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
UserName         : bab

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Untuk detail API, lihat ListUsersdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil koleksi pengguna saat ini Akun AWS.

```
Get-IAMUserList
```

Output:

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
```

```
      UserName         : bab

      Arn              : arn:aws:iam::123456789012:user/David
      CreateDate       : 12/10/2014 3:39:27 PM
      PasswordLastUsed : 3/19/2015 8:44:04 AM
      Path             : /
      UserId           : Y4FKWQCXTA52QEXAMPLE3
      UserName         : David
```

- Untuk detail API, lihat ListUsersdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Untuk detail API, lihat ListUsersdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
# Lists all users in the AWS account
#
# @return [Array<Aws::IAM::Types::User>] An array of user objects
def list_users
  users = []
  @iam_client.list_users.each_page do |page|
    page.users.each do |user|
      users << user
    end
  end
  users
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing users: #{e.message}")
  []
end
```

- Untuk detail API, lihat ListUsersdi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
pub async fn list_users(
```

```
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Untuk detail API, lihat ListUsersreferensi AWS SDK for Rust API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        oo_result = lo_iam->listusers( ).
        MESSAGE 'Retrieved user list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing users.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListUsersdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []

        // Use "Paginated" to get all the users.
        // This lets the SDK handle the 'isTruncated' in "ListUsersOutput".
        let input = ListUsersInput()
        let output = client.listUsersPaginated(input: input)

        do {
            for try await page in output {
                guard let users = page.users else {
                    continue
                }
                for user in users {
                    if let id = user.userId, let name = user.userName {
                        userList.append(MyUserRecord(id: id, name: name))
                    }
                }
            }
        }
        catch {
            print("ERROR: listUsers:", dump(error))
            throw error
        }
       return userList
    }
```

- Untuk detail API, lihat referensi ListUsers AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListVirtualMfaDevices`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ListVirtualMfaDevices`.

CLI

AWS CLI

Untuk daftar perangkat MFA virtual

`list-virtual-mfa-devices`Perintah berikut mencantumkan perangkat MFA virtual yang telah dikonfigurasi untuk akun saat ini.

```
aws iam list-virtual-mfa-devices
```

Output:

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mengaktifkan perangkat otentikasi multi-faktor virtual (MFA) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat ListVirtualMfaDevicesdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengambil koleksi perangkat MFA virtual yang ditetapkan untuk pengguna di AWS akun. **`User`**Properti masing-masing adalah objek dengan detail pengguna IAM tempat perangkat ditugaskan.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Output:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Untuk detail API, lihat ListVirtualMfaDevicesdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengambil koleksi perangkat MFA virtual yang ditetapkan untuk pengguna di AWS akun. **`User`**Properti masing-masing adalah objek dengan detail pengguna IAM tempat perangkat ditugaskan.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Output:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
```

```
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Untuk detail API, lihat ListVirtualMfaDevicesdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutGroupPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutGroupPolicy`.

CLI

AWS CLI

Untuk menambahkan kebijakan ke grup

`put-group-policy`Perintah berikut menambahkan kebijakan ke grup IAM bernama`Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai dokumen JSON dalam AdminPolicy.jsonfile. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk informasi selengkapnya, lihat Mengelola kebijakan IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat PutGroupPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **`AppTesterPolicy`** dan menyematkannya dalam grup IAM. **`AppTesters`** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON datang file**`apptesterpolicy.json`**. Perhatikan bahwa Anda harus menggunakan **`-Raw`** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Untuk detail API, lihat PutGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kebijakan inline bernama **`AppTesterPolicy`** dan menyematkannya dalam grup IAM. **`AppTesters`** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON datang file**`apptesterpolicy.json`**. Perhatikan bahwa Anda harus menggunakan **`-Raw`** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Untuk detail API, lihat PutGroupPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutRolePermissionsBoundary`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutRolePermissionsBoundary`.

CLI

AWS CLI

Contoh 1: Untuk menerapkan batas izin berdasarkan kebijakan kustom ke peran IAM

`put-role-permissions-boundary`Contoh berikut menerapkan kebijakan kustom yang dinamai `intern-boundary` sebagai batas izin untuk peran IAM yang ditentukan.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

Perintah ini tidak menghasilkan output.

Contoh 2: Untuk menerapkan batas izin berdasarkan kebijakan AWS terkelola ke peran IAM

`put-role-permissions-boundary`Contoh berikut menerapkan `PowerUserAccess` kebijakan AWS terkelola sebagai batas izin untuk peran IAM yang ditentukan.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat PutRolePermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk Peran IAM. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat PutRolePermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk Peran IAM. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat PutRolePermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutRolePolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutRolePolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan gudang data tanpa server
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan klaster gudang data yang disediakan
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
```

```
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat PutRolePolicydi Referensi AWS SDK untuk .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);
```

```
    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat PutRolePolicydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk melampirkan kebijakan izin ke peran IAM

`put-role-policy`Perintah berikut menambahkan kebijakan izin ke peran bernama`Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai dokumen JSON dalam AdminPolicy.jsonfile. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk melampirkan kebijakan kepercayaan ke suatu peran, gunakan `update-assume-role-policy` perintah.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat PutRolePolicydi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::amzn-s3-demo-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Untuk detail API, lihat PutRolePolicydi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **FedTesterRolePolicy** dan menyematkannya dalam peran IAM. **FedTesterRole** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file**FedTesterPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Untuk detail API, lihat PutRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kebijakan inline bernama **FedTesterRolePolicy** dan menyematkannya dalam peran IAM. **FedTesterRole** Jika kebijakan inline dengan

nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file**`FedTesterPolicy.json`**. Perhatikan bahwa Anda harus menggunakan **`-Raw`** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Untuk detail API, lihat PutRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutUserPermissionsBoundary`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`PutUserPermissionsBoundary`.

CLI

AWS CLI

Contoh 1: Untuk menerapkan batas izin berdasarkan kebijakan kustom ke pengguna IAM

`put-user-permissions-boundary`Contoh berikut menerapkan kebijakan kustom bernama `intern-boundary` sebagai batas izin untuk pengguna IAM yang ditentukan.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Perintah ini tidak menghasilkan output.

Contoh 2: Untuk menerapkan batas izin berdasarkan kebijakan AWS terkelola ke pengguna IAM

`put-user-permissions-boundary`Contoh berikut menerapkan pollicy AWS terkelola bernama `PowerUserAccess` sebagai batas izin untuk pengguna IAM tertentu.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Perintah ini tidak menghasilkan output.

Untuk informasi lebih lanjut, lihat Menambahkan dan menghapus izin identitas IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat PutUserPermissionsBoundarydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk pengguna. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat PutUserPermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menunjukkan cara mengatur batas Izin untuk pengguna. Anda dapat menetapkan Kebijakan AWS terkelola atau Kebijakan khusus sebagai batas izin.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Untuk detail API, lihat PutUserPermissionsBoundarydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutUserPolicy`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutUserPolicy`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Pelajari dasar-dasarnya

CLI

AWS CLI

Untuk melampirkan kebijakan ke pengguna IAM

`put-user-policy`Perintah berikut melampirkan kebijakan ke pengguna IAM bernama. `Bob`

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan didefinisikan sebagai dokumen JSON dalam AdminPolicy.jsonfile. (Nama file dan ekstensi tidak memiliki signifikansi.)

Untuk informasi lebih lanjut, lihat Menambahkan dan menghapus izin identitas IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat PutUserPolicydi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Untuk detail API, lihat PutUserPolicydi Referensi AWS SDK untuk Go API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini membuat kebijakan inline bernama **EC2AccessPolicy** dan menyematkannya di pengguna IAM. **Bob** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file**EC2AccessPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Untuk detail API, lihat PutUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini membuat kebijakan inline bernama **EC2AccessPolicy** dan menyematkannya di pengguna IAM. **Bob** Jika kebijakan inline dengan nama yang sama sudah ada, maka itu akan ditimpa. Konten kebijakan JSON berasal dari file**EC2AccessPolicy.json**. Perhatikan bahwa Anda harus menggunakan **-Raw** parameter untuk berhasil memproses konten file JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Untuk detail API, lihat PutUserPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates an inline policy for a specified user.
 # @param username [String] The name of the IAM user.
 # @param policy_name [String] The name of the policy to create.
 # @param policy_document [String] The JSON policy document.
 # @return [Boolean]
 def create_user_policy(username, policy_name, policy_document)
   @iam_client.put_user_policy({
                                 user_name: username,
                                 policy_name: policy_name,
                                 policy_document: policy_document
                               })
   @logger.info("Policy #{policy_name} created for user #{username}.")
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   false
 end
```

- Untuk detail API, lihat PutUserPolicydi Referensi AWS SDK untuk Ruby API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSIAM
import AWSS3


    func putUserPolicy(policyDocument: String, policyName: String, user:
 IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            print("ERROR: putUserPolicy:", dump(error))
            throw error
        }
    }
```

- Untuk detail API, lihat referensi PutUserPolicy AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`RemoveClientIdFromOpenIdConnectProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Untuk menghapus ID klien yang ditentukan dari daftar ID klien yang terdaftar untuk penyedia IAM OpenID Connect yang ditentukan

Contoh ini menghapus ID klien `My-TestApp-3` dari daftar ID klien yang terkait dengan penyedia IAM OIDC yang ARN. `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat RemoveClientIdFromOpenIdConnectProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus ID klien **`My-TestApp-3`** dari daftar ID klien yang terkait dengan penyedia IAM OIDC yang ARN-nya. **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Untuk detail API, lihat RemoveClientIdFromOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus ID klien **My-TestApp-3** dari daftar ID klien yang terkait dengan penyedia IAM OIDC yang ARN-nya. **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Untuk detail API, lihat RemoveClientIdFromOpenIdConnectProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **RemoveRoleFromInstanceProfile**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveRoleFromInstanceProfile`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan streaming terkelola
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

CLI

AWS CLI

Untuk menghapus peran dari profil instance

`remove-role-from-instance-profile`Perintah berikut menghapus peran bernama `Test-Role` dari profil instance bernama`ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

Untuk informasi lebih lanjut, lihat Menggunakan profil instance dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat RemoveRoleFromInstanceProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus peran bernama **MyNewRole** dari profil instans EC2 bernama. **MyNewRole** Profil instance yang dibuat di konsol IAM selalu memiliki nama yang sama dengan peran, seperti dalam contoh ini. Jika Anda membuatnya di API atau CLI, maka mereka dapat memiliki nama yang berbeda.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Untuk detail API, lihat RemoveRoleFromInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus peran bernama **MyNewRole** dari profil instans EC2 bernama. **MyNewRole** Profil instance yang dibuat di konsol IAM selalu memiliki nama yang sama dengan peran, seperti dalam contoh ini. Jika Anda membuatnya di API atau CLI, maka mereka dapat memiliki nama yang berbeda.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Untuk detail API, lihat RemoveRoleFromInstanceProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **RemoveUserFromGroup**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`RemoveUserFromGroup`.

CLI

AWS CLI

Untuk menghapus pengguna dari grup IAM

`remove-user-from-group`Perintah berikut menghapus nama pengguna Bob dari grup IAM bernama`Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menambahkan dan menghapus pengguna dalam grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat RemoveUserFromGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus pengguna IAM **`Bob`** dari grup**`Testers`**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Contoh 2: Contoh ini menemukan grup yang pengguna IAM **`Theresa`** adalah anggota, dan kemudian dihapus **`Theresa`** dari grup tersebut.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Contoh 3: Contoh ini menunjukkan cara alternatif untuk menghapus pengguna IAM **`Bob`** dari **`Testers`** grup.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Untuk detail API, lihat RemoveUserFromGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus pengguna IAM **Bob** dari grup**Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Contoh 2: Contoh ini menemukan grup yang pengguna IAM **Theresa** adalah anggota, dan kemudian dihapus **Theresa** dari grup tersebut.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Contoh 3: Contoh ini menunjukkan cara alternatif untuk menghapus pengguna IAM **Bob** dari **Testers** grup.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Untuk detail API, lihat RemoveUserFromGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **ResyncMfaDevice**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`ResyncMfaDevice`.

CLI

AWS CLI

Untuk menyinkronkan perangkat MFA

`resync-mfa-device`Contoh berikut menyinkronkan perangkat MFA yang terkait dengan pengguna IAM Bob dan ARN-nya dengan program autentikator yang `arn:aws:iam::123456789012:mfa/BobsMFADevice` menyediakan dua kode otentikasi.

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menggunakan autentikasi multifaktor (MFA) dalam AWS dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat ResyncMfaDevicedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menyinkronkan perangkat MFA yang terkait dengan pengguna IAM **Bob** dan ARN-nya dengan program autentikator yang **arn:aws:iam::123456789012:mfa/bob** menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Contoh 2: Contoh ini menyinkronkan perangkat IAM MFA yang dikaitkan dengan pengguna **Theresa** IAM dengan perangkat fisik yang memiliki nomor seri **ABCD12345678** dan yang menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Untuk detail API, lihat ResyncMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menyinkronkan perangkat MFA yang terkait dengan pengguna IAM **Bob** dan ARN-nya dengan program autentikator yang **arn:aws:iam::123456789012:mfa/bob** menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Contoh 2: Contoh ini menyinkronkan perangkat IAM MFA yang dikaitkan dengan pengguna **Theresa** IAM dengan perangkat fisik yang memiliki nomor seri **ABCD12345678** dan yang menyediakan dua kode otentikasi.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Untuk detail API, lihat ResyncMfaDevicedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **SetDefaultPolicyVersion**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`SetDefaultPolicyVersion`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kebijakan
- Kembalikan versi kebijakan

CLI

AWS CLI

Untuk menetapkan versi tertentu dari kebijakan yang ditentukan sebagai versi default kebijakan.

Contoh ini menetapkan v2 versi kebijakan yang ARN-nya `arn:aws:iam::123456789012:policy/MyPolicy` sebagai versi aktif default.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Untuk informasi selengkapnya, lihat Kebijakan dan izin di IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat SetDefaultPolicyVersiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan **v2** versi kebijakan yang ARN-nya **arn:aws:iam::123456789012:policy/MyPolicy** sebagai versi aktif default.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Untuk detail API, lihat SetDefaultPolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menetapkan **v2** versi kebijakan yang ARN-nya **arn:aws:iam::123456789012:policy/MyPolicy** sebagai versi aktif default.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Untuk detail API, lihat SetDefaultPolicyVersiondi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
TRY.
    lo_iam->setdefaultpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Default policy version set successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'Invalid input provided.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat SetDefaultPolicyVersiondi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`TagRole`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`TagRole`.

CLI

AWS CLI

Untuk menambahkan tag ke peran

`tag-role`Perintah berikut menambahkan tag dengan nama Departemen ke peran yang ditentukan.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat TagRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag ke Peran dalam Layanan Manajemen Identitas

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat TagRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menambahkan tag ke Peran dalam Layanan Manajemen Identitas

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat TagRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`TagUser`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`TagUser`.

CLI

AWS CLI

Untuk menambahkan tag ke pengguna

`tag-user`Perintah berikut menambahkan tag dengan Departemen terkait untuk pengguna tertentu.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat TagUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menambahkan tag ke Pengguna di Layanan Manajemen Identitas

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat TagUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menambahkan tag ke Pengguna di Layanan Manajemen Identitas

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Untuk detail API, lihat TagUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UntagRole`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UntagRole`.

CLI

AWS CLI

Untuk menghapus tag dari peran

`untag-role`Perintah berikut menghapus tag apa pun dengan nama kunci 'Departemen' dari peran yang ditentukan.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UntagRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag dari peran bernama "MyRoleName" dengan kunci tag sebagai “abac”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Untuk detail API, lihat UntagRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus tag dari peran bernama "MyRoleName" dengan kunci tag sebagai “abac”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Untuk detail API, lihat UntagRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UntagUser`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UntagUser`.

CLI

AWS CLI

Untuk menghapus tag dari pengguna

`untag-user`Perintah berikut menghapus tag apa pun dengan nama kunci 'Departemen' dari pengguna yang ditentukan.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Menandai sumber daya IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UntagUserdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menghapus tag dari pengguna bernama “joe” dengan kunci tag sebagai “abac” dan “xyzw”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Untuk detail API, lihat UntagUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menghapus tag dari pengguna bernama “joe” dengan kunci tag sebagai “abac” dan “xyzw”. Untuk menghapus beberapa tag, berikan daftar kunci tag yang dipisahkan koma.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Untuk detail API, lihat UntagUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateAccessKey`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAccessKey`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kelola kunci akses

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
#############################################################################
# function iam_update_access_key
#
# This function can activate or deactivate an IAM access key for the specified
 IAM user.
#
```

```
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to update.
#       -a            -- Activate the selected access key.
#       -d            -- Deactivate the selected access key.
#
# Example:
#       # To deactivate the selected access key for IAM user Bob
#       iam_update_access_key -u Bob -k AKIAIOSFODNN7EXAMPLE -d
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_update_access_key() {
  local user_name access_key status response
  local option OPTARG # Required to use getopts command in a function.
  local activate_flag=false deactivate_flag=false

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_update_access_key"
    echo "Updates the status of an AWS Identity and Access Management (IAM)
 access key for the specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to update."
    echo "  -a              Activate the access key."
    echo "  -d              Deactivate the access key."
    echo ""
  }

  # Retrieve the calling parameters.
    while getopts "u:k:adh" option; do
      case "${option}" in
        u) user_name="${OPTARG}" ;;
        k) access_key="${OPTARG}" ;;
        a) activate_flag=true ;;
        d) deactivate_flag=true ;;
        h)
          usage
          return 0
          ;;
        \?)
          echo "Invalid parameter"
```

```
          usage
          return 1
          ;;
      esac
    done
    export OPTIND=1

  # Validate input parameters
    if [[ -z "$user_name" ]]; then
      errecho "ERROR: You must provide a username with the -u parameter."
      usage
      return 1
    fi

    if [[ -z "$access_key" ]]; then
      errecho "ERROR: You must provide an access key with the -k parameter."
      usage
      return 1
    fi

    # Ensure that only -a or -d is specified
    if [[ "$activate_flag" == true && "$deactivate_flag" == true ]]; then
      errecho "ERROR: You cannot specify both -a (activate) and -d (deactivate)
at the same time."
      usage
      return 1
    fi

    # If neither -a nor -d is provided, return an error
    if [[ "$activate_flag" == false && "$deactivate_flag" == false ]]; then
      errecho "ERROR: You must specify either -a (activate) or -d (deactivate)."
      usage
      return 1
    fi

    # Determine the status based on the flag
    if [[ "$activate_flag" == true ]]; then
      status="Active"
    elif [[ "$deactivate_flag" == true ]]; then
      status="Inactive"
    fi

    iecho "Parameters:\n"
    iecho "    Username:   $user_name"
```

```
    iecho "    Access key: $access_key"
    iecho "    New status: $status"
    iecho ""

    # Update the access key status
    response=$(aws iam update-access-key \
      --user-name "$user_name" \
      --access-key-id "$access_key" \
      --status "$status" 2>&1)

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log $error_code
      errecho "ERROR: AWS reports update-access-key operation failed.\n$response"
      return 1
    fi

    iecho "update-access-key response: $response"
    iecho

    return 0
}
```

- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengaktifkan atau menonaktifkan kunci akses untuk pengguna IAM

`update-access-key`Perintah berikut menonaktifkan kunci akses yang ditentukan (ID kunci akses dan kunci akses rahasia) untuk pengguna IAM bernama. Bob

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Perintah ini tidak menghasilkan output.

Menonaktifkan kunci berarti tidak dapat digunakan untuk akses terprogram ke. AWS Namun, kuncinya masih tersedia dan dapat diaktifkan kembali.

Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
```

```
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
 String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
```

```
            System.out.printf("Successfully updated the status of access key %s
 to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui kunci akses.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
```

```
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateAccessKeydi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengubah status kunci akses **AKIAIOSFODNN7EXAMPLE** untuk pengguna IAM bernama**Bob**. **Inactive**

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Untuk detail API, lihat UpdateAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengubah status kunci akses **AKIAIOSFODNN7EXAMPLE** untuk pengguna IAM bernama**Bob**. **Inactive**

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Untuk detail API, lihat UpdateAccessKeydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def update_key(user_name, key_id, activate):
```

```
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
key_id
        )
        raise
```

- Untuk detail API, lihat UpdateAccessKeydi AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->updateaccesskey(
          iv_accesskeyid = iv_access_key_id
```

```
          iv_status = iv_status
          iv_username = iv_user_name ).
        MESSAGE 'Access key updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Access key or user does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat UpdateAccessKeydi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateAccountPasswordPolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Untuk mengatur atau mengubah kebijakan kata sandi akun saat ini

`update-account-password-policy`Perintah berikut menetapkan kebijakan kata sandi untuk memerlukan panjang minimum delapan karakter dan memerlukan satu atau lebih angka dalam kata sandi.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Perintah ini tidak menghasilkan output.

Perubahan pada kebijakan kata sandi akun memengaruhi kata sandi baru apa pun yang dibuat untuk pengguna IAM di akun. Perubahan kebijakan kata sandi tidak memengaruhi kata sandi yang ada.

Untuk informasi selengkapnya, lihat Menyetel kebijakan kata sandi akun untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateAccountPasswordPolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui kebijakan kata sandi untuk akun dengan pengaturan yang ditentukan. Perhatikan bahwa parameter apa pun yang tidak termasuk dalam perintah tidak dibiarkan tidak dimodifikasi. Sebaliknya, mereka diatur ulang ke nilai default.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Untuk detail API, lihat UpdateAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui kebijakan kata sandi untuk akun dengan pengaturan yang ditentukan. Perhatikan bahwa parameter apa pun yang tidak termasuk dalam perintah tidak dibiarkan tidak dimodifikasi. Sebaliknya, mereka diatur ulang ke nilai default.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Untuk detail API, lihat UpdateAccountPasswordPolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateAssumeRolePolicy`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateAssumeRolePolicy`.

CLI

AWS CLI

Untuk memperbarui kebijakan kepercayaan untuk peran IAM

`update-assume-role-policy`Perintah berikut memperbarui kebijakan kepercayaan untuk peran bernama`Test-Role`.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Perintah ini tidak menghasilkan output.

Kebijakan kepercayaan didefinisikan sebagai dokumen JSON dalam Test-Role-Trust-Policy.jsonfile. (Nama file dan ekstensi tidak memiliki signifikansi.) Kebijakan kepercayaan harus menentukan kepala sekolah.

Untuk memperbarui kebijakan izin untuk peran, gunakan `put-role-policy` perintah.

Untuk informasi selengkapnya, lihat Membuat peran IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateAssumeRolePolicydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui peran IAM yang diberi nama **`ClientRole`** dengan kebijakan kepercayaan baru, yang isinya berasal dari file**`ClientRolePolicy.json`**. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses isi file JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Untuk detail API, lihat UpdateAssumeRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui peran IAM yang diberi nama **`ClientRole`** dengan kebijakan kepercayaan baru, yang isinya berasal dari file**`ClientRolePolicy.json`**. Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses isi file JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Untuk detail API, lihat UpdateAssumeRolePolicydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateGroup`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateGroup`.

CLI

AWS CLI

Untuk mengganti nama grup IAM

`update-group`Perintah berikut mengubah nama grup IAM `Test` menjadi`Test-1`.

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengganti nama grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateGroupdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama grup IAM menjadi. **`Testers` `AppTesters`**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Contoh 2: Contoh ini mengubah jalur grup IAM **`AppTesters`** menjadi**`/Org1/Org2/`**. Ini mengubah ARN untuk grup menjadi. **`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Untuk detail API, lihat UpdateGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengganti nama grup IAM menjadi. **`Testers` `AppTesters`**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Contoh 2: Contoh ini mengubah jalur grup IAM **`AppTesters`** menjadi**`/Org1/Org2/`**. Ini mengubah ARN untuk grup menjadi. **`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Untuk detail API, lihat UpdateGroupdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateLoginProfile`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateLoginProfile`.

CLI

AWS CLI

Untuk memperbarui kata sandi untuk pengguna IAM

`update-login-profile`Perintah berikut membuat password baru untuk pengguna IAM bernama`Bob`.

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Perintah ini tidak menghasilkan output.

Untuk menetapkan kebijakan kata sandi untuk akun, gunakan `update-account-password-policy` perintah. Jika kata sandi baru melanggar kebijakan kata sandi akun, perintah mengembalikan `PasswordPolicyViolation` kesalahan.

Jika kebijakan kata sandi akun memungkinkan mereka untuk, pengguna IAM dapat mengubah kata sandi mereka sendiri menggunakan `change-password` perintah.

Simpan kata sandi di tempat yang aman. Jika kata sandi hilang, itu tidak dapat dipulihkan, dan Anda harus membuat yang baru menggunakan `create-login-profile` perintah.

Untuk informasi selengkapnya, lihat Mengelola kata sandi untuk pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateLoginProfiledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini menetapkan kata sandi sementara baru untuk pengguna IAM**Bob**, dan mengharuskan pengguna untuk mengubah kata sandi saat pengguna masuk berikutnya.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Untuk detail API, lihat UpdateLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini menetapkan kata sandi sementara baru untuk pengguna IAM**Bob**, dan mengharuskan pengguna untuk mengubah kata sandi saat pengguna masuk berikutnya.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Untuk detail API, lihat UpdateLoginProfiledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateOpenIdConnectProviderThumbprint`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Untuk mengganti daftar sidik jari sertifikat server yang ada dengan daftar baru

Contoh ini memperbarui daftar cap jempol sertifikat untuk penyedia OIDC yang ARN-nya menggunakan sidik jari baru. `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Jika penemuan penyedia OIDC dan titik akhir JWKS Anda menggunakan sertifikat yang berbeda, sertakan cap jempol untuk kedua titik akhir dalam nilai. `--thumbprint-list`

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Membuat penyedia identitas OpenID Connect (OIDC) di Panduan Pengguna IAM.AWS

- Untuk detail API, lihat UpdateOpenIdConnectProviderThumbprintdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui daftar cap jempol sertifikat untuk penyedia OIDC yang ARN-nya menggunakan sidik jari baru. **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** Penyedia OIDC membagikan nilai baru ketika sertifikat yang terkait dengan penyedia berubah.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Untuk detail API, lihat UpdateOpenIdConnectProviderThumbprintdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui daftar cap jempol sertifikat untuk penyedia OIDC yang ARN-nya menggunakan sidik jari baru. **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** Penyedia OIDC membagikan nilai baru ketika sertifikat yang terkait dengan penyedia berubah.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Untuk detail API, lihat UpdateOpenIdConnectProviderThumbprintdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateRole`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateRole`.

CLI

AWS CLI

Untuk mengubah deskripsi peran IAM atau durasi sesi

`update-role`Perintah berikut mengubah deskripsi peran IAM `production-role` menjadi `Main production role` dan menetapkan durasi sesi maksimum menjadi 12 jam.

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateRoledi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi peran dan nilai durasi sesi maksimum (dalam detik) yang sesi peran dapat diminta.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Untuk detail API, lihat UpdateRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui deskripsi peran dan nilai durasi sesi maksimum (dalam detik) yang sesi peran dapat diminta.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Untuk detail API, lihat UpdateRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateRoleDescription`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateRoleDescription`.

CLI

AWS CLI

Untuk mengubah deskripsi peran IAM

`update-role`Perintah berikut mengubah deskripsi peran IAM `production-role` menjadi`Main production role`.

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Output:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
```

```
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

Untuk informasi selengkapnya, lihat Memodifikasi peran dalam Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateRoleDescriptiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui deskripsi peran IAM di akun Anda.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Untuk detail API, lihat UpdateRoleDescriptiondi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui deskripsi peran IAM di akun Anda.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Untuk detail API, lihat UpdateRoleDescriptiondi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateSamlProvider`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateSamlProvider`.

CLI

AWS CLI

Untuk memperbarui dokumen metadata untuk penyedia SAFL yang ada

Contoh ini memperbarui penyedia SAFL di IAM yang ARN-nya `arn:aws:iam::123456789012:saml-provider/SAMLADFS` dengan dokumen metadata SAFL baru dari file tersebut. `SAMLMetaData.xml`

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Output:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Untuk informasi selengkapnya, lihat Membuat penyedia identitas IAM SALL di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateSamlProviderdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui penyedia SAFL di IAM yang ARN-nya **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** dengan dokumen metadata SAFL baru dari file. **`SAMLMetaData.xml`** Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses isi file JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Untuk detail API, lihat UpdateSamlProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui penyedia SAFL di IAM yang ARN-nya **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** dengan dokumen metadata SAFL baru dari file. **`SAMLMetaData.xml`** Perhatikan bahwa Anda harus menggunakan parameter **`-Raw`** switch untuk berhasil memproses isi file JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Untuk detail API, lihat UpdateSamlProviderdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateServerCertificate`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateServerCertificate`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);
```

```
    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Untuk detail API, lihat UpdateServerCertificatedi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengubah jalur atau nama sertifikat server di AWS akun Anda

`update-server-certificate`Perintah berikut mengubah nama sertifikat dari `myServerCertificate` menjadi`myUpdatedServerCertificate`. Itu juga mengubah jalur ke `/cloudfront/` sehingga dapat diakses oleh CloudFront layanan Amazon. Perintah ini tidak menghasilkan output. Anda dapat melihat hasil pembaruan dengan menjalankan `list-server-certificates` perintah.

```
aws-iam update-server-certificate \
```

```
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengelola sertifikat server di IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui sertifikat server.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateServerCertificatedi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateServerCertificatedi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama sertifikat yang dinamai**MyServerCertificate**. **MyRenamedServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Contoh 2: Contoh ini memindahkan sertifikat yang dinamai **MyServerCertificate** ke jalur/Org1/Org2/. Ini mengubah ARN untuk sumber daya menjadi. **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Untuk detail API, lihat UpdateServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengganti nama sertifikat yang dinamai**MyServerCertificate**. **MyRenamedServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Contoh 2: Contoh ini memindahkan sertifikat yang dinamai **MyServerCertificate** ke jalur/Org1/Org2/. Ini mengubah ARN untuk sumber daya menjadi. **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Untuk detail API, lihat UpdateServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Untuk detail API, lihat UpdateServerCertificatedi Referensi AWS SDK untuk Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateSigningCertificate`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateSigningCertificate`.

CLI

AWS CLI

Untuk mengaktifkan atau menonaktifkan sertifikat penandatanganan untuk pengguna IAM

`update-signing-certificate`Perintah berikut menonaktifkan sertifikat penandatanganan yang ditentukan untuk pengguna IAM bernama. Bob

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Untuk mendapatkan ID untuk sertifikat penandatanganan, gunakan `list-signing-certificates` perintah.

Untuk informasi selengkapnya, lihat Mengelola sertifikat penandatanganan di Panduan Pengguna Amazon EC2.

- Untuk detail API, lihat UpdateSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini memperbarui sertifikat yang terkait dengan nama pengguna IAM **Bob** dan yang ID sertifikatnya **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** untuk menandainya sebagai tidak aktif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Untuk detail API, lihat UpdateSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini memperbarui sertifikat yang terkait dengan nama pengguna IAM **Bob** dan yang ID sertifikatnya **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** untuk menandainya sebagai tidak aktif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Untuk detail API, lihat UpdateSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UpdateUser`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UpdateUser`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Buat pengguna read-only dan read-write

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat UpdateUserdi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengubah nama pengguna IAM

`update-user`Perintah berikut mengubah nama pengguna IAM Bob menjadi`Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Mengganti nama grup pengguna IAM di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat UpdateUserdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
```

```
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat UpdateUserdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Perbarui pengguna.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateUserdi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK untuk JavaScript.
- Untuk detail API, lihat UpdateUserdi Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?,
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
```

```
    }
}
```

- Untuk detail API, lihat UpdateUserdi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengganti nama pengguna IAM menjadi. **Bob Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Contoh 2: Contoh ini mengubah jalur Pengguna IAM **Bob** ke**/Org1/Org2/**, yang secara efektif mengubah ARN untuk pengguna. **arn:aws:iam::123456789012:user/Org1/Org2/bob**

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Untuk detail API, lihat UpdateUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengganti nama pengguna IAM menjadi. **Bob Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Contoh 2: Contoh ini mengubah jalur Pengguna IAM **Bob** ke**/Org1/Org2/**, yang secara efektif mengubah ARN untuk pengguna. **arn:aws:iam::123456789012:user/Org1/Org2/bob**

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Untuk detail API, lihat UpdateUserdi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Untuk detail API, lihat UpdateUserdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
  # Updates an IAM user's name
  #
  # @param current_name [String] The current name of the user
  # @param new_name [String] The new name of the user
  def update_user_name(current_name, new_name)
    @iam_client.update_user(user_name: current_name, new_user_name: new_name)
    true
  rescue StandardError => e
    @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
    false
  end
```

- Untuk detail API, lihat UpdateUserdi Referensi AWS SDK untuk Ruby API.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
    TRY.
        lo_iam->updateuser(
          iv_username = iv_user_name
          iv_newusername = iv_new_user_name ).
        MESSAGE 'User updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'New user name already exists.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat UpdateUserdi AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UploadServerCertificate`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UploadServerCertificate`.

CLI

AWS CLI

Untuk mengunggah sertifikat server ke AWS akun Anda

Perintah upload-server-certificate berikut ini mengunggah sertifikat server ke akun Anda. AWS Dalam contoh ini, sertifikat ada di file`public_key_cert_file.pem`, kunci pribadi terkait ada di file`my_private_key.pem`, dan rantai sertifikat yang disediakan oleh otoritas sertifikat (CA) ada di `my_certificate_chain_file.pem` file. Ketika file telah selesai diunggah, itu tersedia dengan nama saya ServerCertificate. Parameter yang dimulai dengan `file://` memberi tahu perintah untuk membaca isi file dan menggunakannya sebagai nilai parameter alih-alih nama file itu sendiri.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Output:

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Untuk informasi selengkapnya, lihat Membuat, Mengunggah, dan Menghapus Sertifikat Server dalam panduan Menggunakan IAM.

- Untuk detail API, lihat UploadServerCertificatedi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "node:path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
```

```
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Untuk detail API, lihat UploadServerCertificatedi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengunggah sertifikat server baru ke akun IAM. File yang berisi badan sertifikat, kunci pribadi, dan (opsional) rantai sertifikat semuanya harus dikodekan PEM. Perhatikan bahwa parameter memerlukan konten sebenarnya dari file daripada nama file. Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses konten file.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
```

```
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat UploadServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengunggah sertifikat server baru ke akun IAM. File yang berisi badan sertifikat, kunci pribadi, dan (opsional) rantai sertifikat semuanya harus dikodekan PEM. Perhatikan bahwa parameter memerlukan konten sebenarnya dari file daripada nama file. Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses konten file.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Output:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Untuk detail API, lihat UploadServerCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UploadSigningCertificate`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`UploadSigningCertificate`.

CLI

AWS CLI

Untuk mengunggah sertifikat penandatanganan untuk pengguna IAM

`upload-signing-certificate`Perintah berikut mengunggah sertifikat penandatanganan untuk pengguna IAM bernama. Bob

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

Output:

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

Sertifikat dalam file bernama certificate.pem dalam format PEM.

Untuk informasi selengkapnya, lihat Membuat dan Mengunggah Sertifikat Penandatanganan Pengguna di panduan Menggunakan IAM.

- Untuk detail API, lihat UploadSigningCertificatedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Contoh ini mengunggah sertifikat X.509 penandatanganan baru dan mengaitkannya dengan nama pengguna IAM. **`Bob`** File yang berisi badan sertifikat dikodekan PEM. **`CertificateBody`**Parameter memerlukan konten sebenarnya dari file sertifikat daripada nama file. Anda harus menggunakan parameter **`-Raw`** sakelar untuk berhasil memproses file.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk detail API, lihat UploadSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Contoh ini mengunggah sertifikat X.509 penandatanganan baru dan mengaitkannya dengan nama pengguna IAM. **Bob** File yang berisi badan sertifikat dikodekan PEM. **CertificateBody**Parameter memerlukan konten sebenarnya dari file sertifikat daripada nama file. Anda harus menggunakan parameter **-Raw** sakelar untuk berhasil memproses file.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Output:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                   YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                   21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                   rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                   nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                   NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                   -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
```

```
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Untuk detail API, lihat UploadSigningCertificatedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Skenario untuk menggunakan IAM AWS SDK

Contoh kode berikut menunjukkan cara menerapkan skenario umum di IAM dengan AWS SDK. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam IAM atau dikombinasikan dengan yang lain. Layanan AWS Setiap skenario menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode.

Skenario menargetkan tingkat pengalaman menengah untuk membantu Anda memahami tindakan layanan dalam konteks.

Contoh

- Membangun dan mengelola layanan tangguh menggunakan AWS SDK
- Konfigurasikan konektivitas layanan kontainer
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Buat API istirahat dengan integrasi proxy fungsi
- Buat pengguna IAM read-only dan read-write menggunakan AWS SDK
- Membuat layanan kontainer untuk instance mesin virtual
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Membuat fungsi tanpa server pertama Anda
- Memulai gudang data tanpa server menggunakan CLI
- Memulai dengan manajemen konfigurasi
- Memulai dengan perlindungan perangkat internet of things
- Memulai dengan toko fitur pembelajaran mesin

- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan klaster gudang data yang disediakan
- Memulai dengan orkestrasi alur kerja
- Mengelola kunci akses IAM menggunakan AWS SDK
- Mengelola kebijakan IAM menggunakan AWS SDK
- Mengelola peran IAM menggunakan AWS SDK
- Mengelola akun IAM Anda menggunakan AWS SDK
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Memungkinkan AWS Compute Optimizer Fitur otomatisasi untuk menerapkan tindakan yang disarankan
- Kebijakan untuk mengaktifkan Otomasi di seluruh organisasi Anda
- Kebijakan untuk mengaktifkan Otomasi untuk akun Anda
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening
- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi
- Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening
- Kebijakan untuk memberikan izin peran terkait layanan untuk Otomatisasi Pengoptimalan Komputasi
- Kembalikan versi kebijakan IAM menggunakan AWS SDK
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Mengatur Attribute-Based Access Control untuk DynamoDB menggunakan AWS Command Line Interface v2
- Menyiapkan manajemen sistem
- Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server
- Bekerja dengan DynamoDB Streams dan menggunakan Time-to-Live AWS Command Line Interface v2

- Bekerja dengan API Pembuat Kebijakan IAM menggunakan AWS SDK

Membangun dan mengelola layanan tangguh menggunakan AWS SDK

Contoh kode berikut menunjukkan cara membuat layanan web dengan beban seimbang yang mengembalikan rekomendasi buku, film, dan lagu. Contoh ini menunjukkan cara layanan tersebut merespons kegagalan, serta cara merestrukturisasi layanan agar lebih tangguh ketika terjadi kegagalan.

- Menggunakan grup Amazon EC2 Auto Scaling untuk membuat instans Amazon Elastic Compute Cloud (Amazon EC2) berdasarkan templat peluncuran dan menyimpan sejumlah instans dalam rentang yang ditentukan.
- Menangani dan mendistribusikan permintaan HTTP dengan Elastic Load Balancing.
- Memantau kondisi instans dalam grup Auto Scaling dan meneruskan permintaan hanya ke instans yang sehat.
- Menjalankan server web Python pada setiap instans EC2 untuk menangani permintaan HTTP. Server web merespons dengan memberikan rekomendasi dan melakukan pemeriksaan kondisi.
- Menyimulasikan layanan yang direkomendasikan dengan tabel Amazon DynamoDB.
- Kontrol respons server web terhadap permintaan dan pemeriksaan kesehatan dengan memperbarui AWS Systems Manager parameter.

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
```

```
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);
```

```
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
```

```
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
```

```
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
```

```
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
```

```
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");
```

```
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
```

```
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
```

```
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );
```

```
        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
 resources? (y/n) "))
        {
            await
 _elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
 _elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
 _autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
 _autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
 _autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Membuat kelas yang menggabungkan tindakan Penskalaan Otomatis dan Amazon EC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
```

```
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;
    private readonly ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;
```

```
        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
```

```
                                    "\"Service\": [" +
                                        "\"ec2.amazonaws.com\"" +
                                    "]" +
                                    "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }
```

```
        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
```

```
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
```

```
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        try
        {
            await CreateKeyPair(_keyPairName);
            await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
                _instanceProfileName, instancePolicyPath);

            var startServerText = await File.ReadAllTextAsync(startupScriptPath);
            var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);
```

```
            var amiLatest = await _amazonSsm.GetParameterAsync(
                new GetParameterRequest() { Name = _amiParam });
            var amiId = amiLatest.Parameter.Value;
            var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
                new CreateLaunchTemplateRequest()
                {
                    LaunchTemplateName = _launchTemplateName,
                    LaunchTemplateData = new RequestLaunchTemplateData()
                    {
                        InstanceType = _instanceType,
                        ImageId = amiId,
                        IamInstanceProfile =
                            new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                            {
                                Name = _instanceProfileName
                            },
                        KeyName = _keyPairName,
                        UserData = System.Convert.ToBase64String(plainTextBytes)
                    }
                });
            return launchTemplateResponse.LaunchTemplate;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
            throw;
        }
    }
```

```
    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
 z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
 availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
 zones.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
```

```
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
```

```
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }
```

```
            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }
```

```
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
```

```
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
```

```
   public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
   {
       try
       {
           var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
               new DescribeIamInstanceProfileAssociationsRequest()
               {
                   Filters = new List<Amazon.EC2.Model.Filter>()
                   {
                       new("instance-id", new List<string>() { instanceId })
                   },
               });
           return response.IamInstanceProfileAssociations[0];
       }
       catch (AmazonEC2Exception ec2Exception)
       {
           if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
           {
               _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
           }

           throw;
       }
       catch (Exception ex)
       {
           _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
           throw;
       }
   }

   /// <summary>
   /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
   /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
   /// used to restart the Python web server.
   /// </summary>
   /// <param name="instanceId">The Id of the instance to update.</param>
   /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
```

```
   /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
   /// <returns>Async task.</returns>
   public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
   {
       try
       {
           await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
               new ReplaceIamInstanceProfileAssociationRequest()
               {
                   AssociationId = associationId,
                   IamInstanceProfile = new IamInstanceProfileSpecification()
                   {
                       Name = credsProfileName
                   }
               });
           // Allow time before resetting.
           Thread.Sleep(25000);

           await _amazonEc2.RebootInstancesAsync(
               new RebootInstancesRequest(new List<string>() { instanceId }));
           Thread.Sleep(25000);
           var instanceReady = false;
           var retries = 5;
           while (retries-- > 0 && !instanceReady)
           {
               var instancesPaginator =
                   _amazonSsm.Paginators.DescribeInstanceInformation(
                       new DescribeInstanceInformationRequest());
               // Get the entire list using the paginator.
               await foreach (var instance in
instancesPaginator.InstanceInformationList)
               {
                   instanceReady = instance.InstanceId == instanceId;
                   if (instanceReady)
                   {
                       break;
                   }
               }
           }
           Console.WriteLine("Waiting for instance to be running.");
           await WaitForInstanceState(instanceId, InstanceStateName.Running);
           Console.WriteLine("Instance ready.");
```

```
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
```

```
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
```

```
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
```

```
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
```

```
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
```

```
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
```

```
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();
```

```
    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
```

```
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }
```

```
    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
 subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
```

```
   /// <param name="subnetIds">Subnets for the load balancer.</param>
   /// <param name="targetGroup">Target group for forwarded requests.</param>
   /// <returns>The new LoadBalancer object.</returns>
   public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
   {
       var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
           new CreateLoadBalancerRequest()
           {
               Name = name,
               Subnets = subnetIds
           });
       var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

       // Wait for load balancer to be available.
       var loadBalancerReady = false;
       while (!loadBalancerReady)
       {
           try
           {
               var describeResponse =
                   await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                       new DescribeLoadBalancersRequest()
                       {
                           Names = new List<string>() { name }
                       });

               var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

               loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
           }
           catch (LoadBalancerNotFoundException)
           {
               loadBalancerReady = false;
           }
           Thread.Sleep(10000);
       }
       // Create the listener.
       await _amazonElasticLoadBalancingV2.CreateListenerAsync(
           new CreateListenerRequest()
```

```
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
```

```
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
```

```
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
targetArn });
                Console.WriteLine($"Deleted load balancing target group
{groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
```

```
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
```

```
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
```

```
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;
```

```
    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk .NET .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup

- CreateInstanceProfile
- CreateLaunchTemplate
- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
```

```
        System.out.println("Welcome to the demonstration of How to Build and
Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
```

```
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);
```

```
        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
```

```
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }
```

```
            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);
```

```
        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
```

```
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
```

```
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
```

```
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
```

```
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
```

```
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Membuat kelas yang menggabungkan tindakan Penskalaan Otomatis dan Amazon EC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }
```

```
    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }
```

```
   /**
    * Replaces the profile associated with a running instance. After the profile
is
    * replaced, the instance is rebooted to ensure that it uses the new profile.
    * When
    * the instance is ready, Systems Manager is used to restart the Python web
    * server.
    */
   public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
           throws InterruptedException {
       // Create an IAM instance profile specification.
       software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
               .builder()
               .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                             // name.
               .build();

       // Replace the IAM instance profile association for the EC2 instance.
       ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
               .builder()
               .iamInstanceProfile(iamInstanceProfile)
               .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
               .build();

       try {
           getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
           // Handle the response as needed.
       } catch (Ec2Exception e) {
           // Handle exceptions, log, or report the error.
           System.err.println("Error: " + e.getMessage());
       }
       System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
               newInstanceProfileName);
       TimeUnit.SECONDS.sleep(15);
       boolean instReady = false;
       int tries = 0;
```

```
        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
```

```
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);
```

```
            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();
```

```
getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
       System.out.println(groupName + " was deleted.");
   }

   /*
    * Verify the default security group of the specified VPC allows ingress from
    * this
    * computer. This can be done by allowing ingress from this computer's IP
    * address. In some situations, such as connecting from a corporate network,
you
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);
```

```
            for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }
```

```
   /*
    * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
    * Scaling group.
    * The target group specifies how the load balancer forward requests to the
    * instances
    * in the group.
    */
   public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
       try {
           AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                   .autoScalingGroupName(asGroupName)
                   .targetGroupARNs(targetGroupARN)
                   .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
           System.out.println("Attached load balancer to " + asGroupName);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()
```

```
.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
                .collect(Collectors.toList());

        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
```

```
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
```

```
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
the policy.

software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
```

```
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
```

```
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
```

```
   public List<TargetHealthDescription> checkTargetHealth(String
targetGroupName) {
       DescribeTargetGroupsRequest targetGroupsRequest =
DescribeTargetGroupsRequest.builder()
               .names(targetGroupName)
               .build();

       DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

       DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
               .build();

       DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
       return healthResponse.targetHealthDescriptions();
   }

   // Gets the HTTP endpoint of the load balancer.
   public String getEndpoint(String lbName) {
       DescribeLoadBalancersResponse res = getLoadBalancerClient()
               .describeLoadBalancers(describe -> describe.names(lbName));
       return res.loadBalancers().get(0).dnsName();
   }

   // Deletes a load balancer.
   public void deleteLoadBalancer(String lbName) {
       try {
           // Use a waiter to delete the Load Balancer.
           DescribeLoadBalancersResponse res = getLoadBalancerClient()
                   .describeLoadBalancers(describe -> describe.names(lbName));
           ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
           DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                   .build();

           getLoadBalancerClient().deleteLoadBalancer(
```

```
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
```

```
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
```

```
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");
```

```
            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;
```

```
    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
such
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
```

```
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
                                    .attributeType(ScalarAttributeType.S)
                                    .build(),
                            AttributeDefinition.builder()
                                    .attributeName("ItemId")
                                    .attributeType(ScalarAttributeType.N)
                                    .build())
                    .keySchema(
                            KeySchemaElement.builder()
                                    .attributeName("MediaType")
                                    .keyType(KeyType.HASH)
                                    .build(),
                            KeySchemaElement.builder()
                                    .attributeName("ItemId")
                                    .keyType(KeyType.RANGE)
                                    .build())
                    .provisionedThroughput(
                            ProvisionedThroughput.builder()
                                    .readCapacityUnits(5L)
                                    .writeCapacityUnits(5L)
                                    .build())
                    .build();

            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
```

```
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
 IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer

- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
// SPDX-License-Identifier: Apache-2.0

import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
```

```
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
 [-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

Menyusun langkah-langkah untuk men-deploy semua sumber daya.

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
```

```
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
```

```
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );
```

```
  return client.send(
    new BatchWriteItemCommand({
      RequestItems: {
        [NAMES.tableName]: recommendations.map((item) => ({
          PutRequest: { Item: item },
        })),
      },
    }),
  );
}),
new ScenarioOutput(
  "populatedTable",
  MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
),
new ScenarioOutput(
  "creatingKeyPair",
  MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioAction("createKeyPair", async () => {
  const client = new EC2Client({});
  const { KeyMaterial } = await client.send(
    new CreateKeyPairCommand({
      KeyName: NAMES.keyPairName,
    }),
  );

  writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
}),
new ScenarioOutput(
  "createdKeyPair",
  MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioOutput(
  "creatingInstancePolicy",
  MESSAGES.creatingInstancePolicy.replace(
    "${INSTANCE_POLICY_NAME}",
    NAMES.instancePolicyName,
  ),
),
new ScenarioAction("createInstancePolicy", async (state) => {
  const client = new IAMClient({});
  const {
    Policy: { Arn },
```

```
  } = await client.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.instancePolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "instance_policy.json"),
      ),
    }),
  );
  state.instancePolicyArn = Arn;
}),
new ScenarioOutput("createdInstancePolicy", (state) =>
  MESSAGES.createdInstancePolicy
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
    .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
),
new ScenarioOutput(
  "creatingInstanceRole",
  MESSAGES.creatingInstanceRole.replace(
    "${INSTANCE_ROLE_NAME}",
    NAMES.instanceRoleName,
  ),
),
new ScenarioAction("createInstanceRole", () => {
  const client = new IAMClient({});
  return client.send(
    new CreateRoleCommand({
      RoleName: NAMES.instanceRoleName,
      AssumeRolePolicyDocument: readFileSync(
        join(ROOT, "assume-role-policy.json"),
      ),
    }),
  );
}),
new ScenarioOutput(
  "createdInstanceRole",
  MESSAGES.createdInstanceRole.replace(
    "${INSTANCE_ROLE_NAME}",
    NAMES.instanceRoleName,
  ),
),
new ScenarioOutput(
  "attachingPolicyToRole",
  MESSAGES.attachingPolicyToRole
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
```

```
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
```

```
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
```

```
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
```

```
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
```

```
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
```

```
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
```

```
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
```

```
    const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
      ({ IpRanges }) =>
        IpRanges.some(
          ({ CidrIp }) =>
            CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
        ),
    )
      .filter(({ IpProtocol }) => IpProtocol === "tcp")
      .filter(({ FromPort }) => FromPort === 80);

    state.myIpRules = myIpRules;
  },
),
new ScenarioOutput(
  "verifiedInboundPort",
  /**
   * @param {{ myIpRules: any[] }} state
   */
  (state) => {
    if (state.myIpRules.length > 0) {
      return MESSAGES.foundIpRules.replace(
        "${IP_RULES}",
        JSON.stringify(state.myIpRules, null, 2),
      );
    }
    return MESSAGES.noIpRules;
  },
),
new ScenarioInput(
  "shouldAddInboundRule",
  /**
   * @param {{ myIpRules: any[] }} state
   */
  (state) => {
    if (state.myIpRules.length > 0) {
      return false;
    }
    return MESSAGES.noIpRules;
  },
  { type: "confirm" },
),
new ScenarioAction(
  "addInboundRule",
  /**
```

```
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
```

```
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

Menyusun langkah-langkah untuk menjalankan demo.

```
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
```

```
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
```

```
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
```

```
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
```

```
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
```

```
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
```

```
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
```

```
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
```

```
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
```

```
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
```

```
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Menyusun langkah-langkah untuk menghancurkan semua sumber daya.

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
```

```
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
```

```
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
```

```
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
```

```
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
```

```
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
```

```
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
```

```
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
```

```
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
```

```
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
    return MESSAGES.detachedSsmOnlyCustomRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
```

```
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
```

```
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
```

```
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
 state,
    ) => {
      const ec2Client = new EC2Client({});

      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
```

```
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
```

```
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk JavaScript .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups

- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Menjalankan skenario interaktif di prompt perintah.

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
```

```
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
```

```
        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
 Zones."
        )
        zones = self.autoscaler.create_autoscaling_group(3)

        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
 balancer.")

        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)

        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
```

```
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                logging.warning(
                    "The default security group for your VPC must allow access
from this computer."
                )
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )
```

```
    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
```

```
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")

    def demo(self) -> None:
        """
        Runs the demonstration, showing how the service responds to different
failure scenarios
        and how a resilient architecture can keep the service running.
        """
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        logging.info("Resetting parameters to starting values for the demo.")
        self.param_helper.reset()

        logging.info(
            "Starting demonstration of the service's resilience under various
failure conditions."
        )
        self.demo_choices()

        logging.info(
            "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info("Sending GET requests will now return failure codes.")
        self.demo_choices()

        logging.info("Switching to static response mode to mitigate failure.")
        self.param_helper.put(self.param_helper.failure_response, "static")
        logging.info("Sending GET requests will now return static responses.")
        self.demo_choices()

        logging.info("Restoring normal operation of the recommendation service.")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
```

```
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
```

```
            "All instances will report as unhealthy, but the service will still
 return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
 Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
 created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
```

```
    """
    Main function to parse arguments and run the appropriate actions for the
demo.
    """
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../scenarios/features/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
```

```
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Membuat kelas yang menggabungkan tindakan Penskalaan Otomatis dan Amazon EC2.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
```

```
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
```

```
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
 definition.
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
            policy_doc = file.read()

        try:
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
 {policy_arn}")
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
```

```
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
        :param aws_managed_policies: A tuple of AWS-managed policy names to
 attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
 PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
 {role_name}.")
```

```
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
```

```
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn


    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
            if not response["IamInstanceProfileAssociations"]:
```

```
                log.info(f"No instance profile found for instance
{instance_id}.")
            profile_data = response["IamInstanceProfileAssociations"][0]
            log.info(f"Retrieved instance profile for instance {instance_id}.")
            return profile_data
        except ClientError as err:
            log.error(
                f"Failed to retrieve instance profile for instance
{instance_id}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidInstanceID.NotFound":
                log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")


    def replace_instance_profile(
        self,
        instance_id: str,
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
```

```
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)

            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)

            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
```

```
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
```

```
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")


    def delete_key_pair(self) -> None:
        """
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
```

```
                                    to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()

            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
```

```
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
                )
            log.error(f"Full error:\n\t{err}")
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
```

```
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
```

```
        else:
            return zones


    def get_instances(self) -> List[str]:
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
```

```
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.
```

```
        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
```

```
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
 to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
 the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
```

```
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
                    "You do not have the necessary permissions to describe VPCs.
 "
                    "Ensure that your AWS IAM user or role has the correct
 permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
 parameters."
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass


    def verify_inbound_port(
        self, vpc: Dict[str, Any], port: int, ip_address: str
    ) -> Tuple[Dict[str, Any], bool]:
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.
```

```
        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specified VPC, and a value
that indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info(f"Found default security group {sec_group['GroupId']}.")

            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info(f"Found inbound rule: {ip_perm}")
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
```

```
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
            )
            if error_code == "InvalidGroupId.Malformed":
                log.error(
                    "The security group ID is malformed. "
                    "Please verify that the security group ID is correct."
```

```
                )
            elif error_code == "InvalidPermission.Duplicate":
                log.error(
                    "The specified rule already exists in the security group. "
                    "Check the existing rules for this security group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        # Ensure that 'zones' is a list, even if None is passed
        if zones is None:
            zones = []
        try:
            paginator = self.ec2_client.get_paginator("describe_subnets")
            page_iterator = paginator.paginate(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )

            subnets = []
            for page in page_iterator:
                subnets.extend(page["Subnets"])

            log.info("Found %s subnets for the specified zones.", len(subnets))
            return subnets
        except ClientError as err:
            log.error(
                f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
            )
            error_code = err.response["Error"]["Code"]
```

```
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    "The specified VPC ID does not exist. "
                    "Please check the VPC ID and try again."
                )
            # Add more error-specific handling as needed
            log.error(f"Full error:\n\t{err}")
```

Membuat kelas yang menggabungkan tindakan Penyeimbangan Beban Elastis.

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing target group. The target group
 specifies how
        the load balancer forwards requests to instances in the group and how
 instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
 times and
        lower thresholds. In production, you might want to decrease the
 sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
```

```
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
```

```
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
```

```
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )


    def create_load_balancer(
        self,
        load_balancer_name: str,
        subnet_ids: List[str],
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param load_balancer_name: The name of the load balancer to create.
        :param subnet_ids: A list of subnets to associate with the load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info(f"Created load balancer '{load_balancer_name}'.")

            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info(
                f"Waiting for load balancer '{load_balancer_name}' to be
available..."
```

```
            )
            waiter.wait(Names=[load_balancer_name])
            log.info(f"Load balancer '{load_balancer_name}' is now available!")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "DuplicateLoadBalancerNameException":
                log.error(
                    f"A load balancer with the name '{load_balancer_name}'
already exists. "
                    "Load balancer names must be unique within the AWS region. "
                    "Please choose a different name and try again."
                )
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer


    def create_listener(
        self,
        load_balancer_name: str,
        target_group: Dict[str, Any],
    ) -> Dict[str, Any]:
        """
        Creates a listener for the specified load balancer that forwards requests
to the
        specified target group.

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
```

```
                            load balancer.
        :return: Data about the newly created listener.
        """
        try:
            # Retrieve the load balancer ARN
            load_balancer_response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            load_balancer_arn = load_balancer_response["LoadBalancers"][0][
                "LoadBalancerArn"
            ]

            # Create the listener
            response = self.elb_client.create_listener(
                LoadBalancerArn=load_balancer_arn,
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
            )
            return response["Listeners"][0]
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
```

```
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_load_balancer(self, load_balancer_name) -> None:
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.
```

```
        :return: The endpoint.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
```

```
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
                    "Ensure the load balancer exists, is in the correct AWS
region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
correct region, "
                    "and ensure it has not been deleted or created in a different
account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

Membuat kelas yang menggunakan DynamoDB untuk menyimulasikan layanan yang direkomendasikan.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
 parameters.

        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as a recommendation service. The table
 has a
        hash key named 'MediaType' that defines the type of media recommended,
 such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
 MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
```

```
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
```

```
            )

    def destroy(self) -> None:
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Membuat kelas yang mengabungkan tindakan Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
```

```
        Initializes the ParameterHelper class with the necessary parameters.

        :param table_name: The name of the DynamoDB table that is used as a
recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
```

```
                log.error(
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Python (Boto3).
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances

- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Konfigurasikan konektivitas layanan kontainer

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat infrastruktur VPC
- Mengatur pencatatan
- Buat cluster ECS
- Konfigurasikan peran IAM
- Buat layanan dengan Service Connect
- Verifikasi penyebaran
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits
```

```
set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}
```

```
# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
```

```
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
```

```
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }
```

```
    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
```

```
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
```

```
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"

    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
```

```
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
```

```
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"

    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
```

```
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================"
    echo "CREATED RESOURCES"
    echo "============================================"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================"
    echo ""
```

```
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
```

```
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
```

```
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups

    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources
```

```
    # Ask user if they want to clean up
    echo ""
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace

- DeleteRole
- DeleteSecurityGroup
- DeleteService (AWS Cloud Map)
- DeleteService (Amazon ECS)
- DeregisterTaskDefinition
- DescribeClusters
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Tunggu (Amazon EC2)
- Tunggu (Amazon ECS)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Buat tugas kontainer untuk tipe peluncuran tanpa server

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat cluster
- Buat definisi tugas
- Buat layanan

- Bersihkan

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
```

```
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "========================================="
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "========================================="

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
```

```
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
```

```
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
```

```
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "==========================================="
    echo "CLEANUP PROCESS"
    echo "==========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."

    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup process..."

        # Step 1: Scale service to 0 tasks first, then delete service
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
 then
            echo ""
            echo "Step 1: Scaling service to 0 tasks..."
            if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
                echo "Waiting for service to stabilize after scaling to 0..."
                execute_command "aws ecs wait services-stable --cluster
 '$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

                echo "Deleting service..."
```

```
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
            else
                echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
            fi
        fi

        # Step 2: Wait a bit for tasks to fully terminate
        echo ""
        echo "Step 2: Waiting for tasks to fully terminate..."
        sleep 15

        # Step 3: Delete cluster
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
" ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
```

```
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "========================================="
        echo "CLEANUP COMPLETED"
        echo "========================================="
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}
```

```
# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "==========================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "==========================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

```
    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================"
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================"

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================"
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================"
```

```
# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
```

```
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "==========================================="
echo "STEP 4: SET UP NETWORKING"
echo "==========================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
```

```
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
```

```
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "============================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "============================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "============================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "============================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"
```

```
# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "========================================="
```

```
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices
  - DescribeSubnets

- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Buat API istirahat dengan integrasi proxy fungsi

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk eksekusi Lambda
- Buat dan terapkan fungsi Lambda
- Buat REST API
- Konfigurasikan integrasi proxy Lambda
- Menerapkan dan menguji API
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
```

```
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
```

```
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration
```

```
# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
```

```
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
```

```
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
```

```
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddPermission
  - AttachRolePolicy

- CreateDeployment
- CreateFunction
- CreateResource
- CreateRestApi
- CreateRole
- DeleteFunction
- DeleteRestApi
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetResources
- GetRestApis
- PutIntegration
- PutMethod

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Buat pengguna IAM read-only dan read-write menggunakan AWS SDK

Contoh kode berikut menunjukkan cara membuat pengguna dan melampirkan kebijakan kepada mereka.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat dua pengguna IAM.
- Lampirkan kebijakan bagi satu pengguna untuk mendapatkan dan meletakkan objek di bucket Amazon S3.

- Lampirkan kebijakan bagi pengguna kedua untuk mendapatkan objek dari bucket.
- Dapatkan izin berbeda ke bucket berdasarkan kredensyal pengguna.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan pengguna IAM.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
```

```
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.
```

```
    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Buat fungsi yang membungkus tindakan kebijakan IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
```

```
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Buat fungsi yang membungkus tindakan kunci akses IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
```

```
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Gunakan fungsi pembungkus untuk membuat pengguna dengan kebijakan berbeda dan menggunakan kredensialnya untuk mengakses bucket Amazon S3.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
```

```
    This demonstration creates two users: one user who can put and get objects in
an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
```

```
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
```

```
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
```

```
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - AttachUserPolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteUser
  - DetachUserPolicy
  - ListUsers
  - UpdateUser

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat layanan kontainer untuk instance mesin virtual

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat cluster ECS
- Buat dan pantau layanan
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
```

```
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
```

```
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi

        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
```

```
       if ! aws ecs deregister-task-definition --task-definition
"$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
           log "WARNING: Failed to deregister task definition
$TASK_DEFINITION_ARN"
           ((cleanup_errors++))
       fi
   fi

   # Terminate EC2 instance
   if [[ -n "${INSTANCE_ID:-}" ]]; then
       log "Terminating EC2 instance: $INSTANCE_ID"
       if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
           log "WARNING: Failed to terminate instance $INSTANCE_ID"
           ((cleanup_errors++))
       else
           log "Waiting for instance to terminate..."
           if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
               log "WARNING: Instance termination wait failed for $INSTANCE_ID"
               ((cleanup_errors++))
           fi
       fi
   fi

   # Delete security group with retry logic
   if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
       log "Deleting security group: $SECURITY_GROUP_ID"
       local retry_count=0
       local max_retries=3

       while [[ $retry_count -lt $max_retries ]]; do
           if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
               log "Successfully deleted security group"
               break
           else
               ((retry_count++))
               if [[ $retry_count -lt $max_retries ]]; then
                   log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                   sleep 10
               else
```

```
                    log "ERROR: Failed to delete security group after
 $max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
```

```
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"

    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"
```

```
    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)
```

```
    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."
```

```
        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
```

```
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."

    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
```

```
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
```

```
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
```

```
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi

    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
```

```
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"

    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
```

```
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "============================================="
```

```
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "========================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
```

```
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
main "$@"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeContainerInstances
  - DescribeInstances

- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetParameters
- GetRole
- ListContainerInstances
- ListServices
- ListTasks
- RegisterTaskDefinition
- RunInstances
- StopTask
- TerminateInstances
- UpdateService
- Tunggu (Amazon EC2)
- Tunggu (Amazon ECS)
- Tunggu (IAM)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat ruang kerja pemantauan terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk ruang kerja Anda
- Buat ruang kerja Grafana
- Konfigurasikan otentikasi
- Konfigurasikan pengaturan opsional
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it

# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
```

```
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
```

```
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"

ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)
```

```
if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi
```

```
# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "=========================================="
echo "WORKSPACE INFORMATION"
echo "=========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "=========================================="

# Step 6: Prompt for cleanup
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
```

```
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done

    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi
```

```
echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace
  - DescribeWorkspace
  - DetachRolePolicy
  - GetCallerIdentity

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat dasbor pemantauan dengan nama fungsi sebagai variabel

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat CloudWatch dasbor
- Tambahkan widget metrik Lambda dengan variabel nama fungsi
- Verifikasi dasbor
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true
```

```
    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi

# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."
```

```
    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi

    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
```

```
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
```

```
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
```

```
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
```

```
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
```

```
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
```

```
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
```

```
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi

    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole

- DetachRolePolicy
- GetCallerIdentity
- GetDashboard
- Memohon
- ListDashboards
- ListFunctions
- PutDashboard

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat fungsi tanpa server pertama Anda

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk Lambda
- Buat kode fungsi
- Buat fungsi Lambda
- Uji fungsi Lambda Anda
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
# views CloudWatch logs, and cleans up all resources.
```

```
#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE -o pipefail

################################################################################
# Setup
################################################################################

UNIQUE_ID=$(head -c 8 /dev/urandom | od -An -tx1 | tr -d ' ')
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
readonly TEMP_DIR
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1

declare -a CREATED_RESOURCES

################################################################################
# Helper functions
################################################################################

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
        local NAME="${RESOURCE#*:}"
```

```
        case "$TYPE" in
            log-group)
                echo "Deleting CloudWatch log group: ${NAME}"
                aws logs delete-log-group \
                    --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete log group ${NAME}."
                ;;
            lambda-function)
                echo "Deleting Lambda function: ${NAME}"
                aws lambda delete-function \
                    --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete Lambda function ${NAME}."
                echo "  Waiting for function deletion to complete..."
                local DELETE_WAIT=0
                while aws lambda get-function --function-name "$NAME" > /dev/null
2>&1; do
                    sleep 2
                    DELETE_WAIT=$((DELETE_WAIT + 2))
                    if [ "$DELETE_WAIT" -ge 60 ]; then
                        echo "  WARNING: Timed out waiting for function
deletion."
                        break
                    fi
                done
                ;;
            iam-role-policy)
                local ROLE_PART="${NAME%%|*}"
                local POLICY_PART="${NAME#*|}"
                echo "Detaching policy from role: ${ROLE_PART}"
                aws iam detach-role-policy \
                    --role-name "$ROLE_PART" \
                    --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
not detach policy from role ${ROLE_PART}."
                ;;
            iam-role)
                echo "Deleting IAM role: ${NAME}"
                aws iam delete-role \
                    --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
IAM role ${NAME}."
                ;;
        esac
    done
```

```
    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "========================================="
    echo "ERROR: Script failed at $1"
    echo "========================================="
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
    local COMMAND="$2"
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
 seconds."
            return 1
        fi
```

```
        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

validate_input() {
    local input="$1"
    local pattern="$2"
    if ! [[ "$input" =~ $pattern ]]; then
        echo "ERROR: Invalid input: $input"
        return 1
    fi
    return 0
}

################################################################################
# Region pre-check
################################################################################

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
 "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi

################################################################################
# Runtime selection
################################################################################

echo ""
echo "========================================="
echo "AWS Lambda - Create Your First Function"
echo "========================================="
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Using default: Python 3.13"
RUNTIME_CHOICE="1"

case "$RUNTIME_CHOICE" in
```

```
    1)
        RUNTIME="python3.13"
        HANDLER="lambda_function.lambda_handler"
        CODE_FILE="lambda_function.py"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'
import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    if not isinstance(event, dict) or 'length' not in event or 'width' not in
 event:
        raise ValueError('Event must contain length and width')
    try:
        length = float(event['length'])
        width = float(event['width'])
        if length < 0 or width < 0:
            raise ValueError('Length and width must be non-negative')
        area = calculate_area(length, width)
        print(f'The area is {area}')
        logger.info(f'CloudWatch logs group: {context.log_group_name}')
        return json.dumps({'area': area})
    except (TypeError, ValueError) as e:
        logger.error(f'Error processing input: {str(e)}')
        raise
def calculate_area(length, width):
    return length * width
PYTHON_EOF
        echo "Selected runtime: Python 3.13"
        ;;
    2)
        RUNTIME="nodejs22.x"
        HANDLER="index.handler"
        CODE_FILE="index.mjs"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'
export const handler = async (event, context) => {
  if (!event || typeof event.length !== 'number' || typeof event.width !==
 'number') {
    throw new Error('Event must contain numeric length and width');
  }
  if (event.length < 0 || event.width < 0) {
    throw new Error('Length and width must be non-negative');
  }
  const area = event.length * event.width;
```

```
  console.log(`The area is ${area}`);
  console.log('CloudWatch log group: ', context.logGroupName);
  return JSON.stringify({area});
};
NODEJS_EOF
        echo "Selected runtime: Node.js 22.x"
        ;;
    *)
        echo "ERROR: Invalid choice. Please enter 1 or 2."
        exit 1
        ;;
esac

# ============================================================================
# Step 1: Create IAM execution role
# ============================================================================

echo ""
echo "==========================================="
echo "Step 1: Create IAM execution role"
echo "==========================================="
echo ""

TRUST_POLICY='{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)

if ! validate_input "$ROLE_OUTPUT" "^arn:aws:iam::[0-9]+:role/"; then
```

```
    echo "ERROR: Failed to create IAM role"
    exit 1
fi

echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

aws iam tag-role \
    --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=lambda-gettingstarted

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

################################################################################
# Step 2: Create Lambda function
################################################################################

echo ""
echo "========================================="
echo "Step 2: Create Lambda function"
echo "========================================="
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR" || exit 1
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1 || {
    echo "ERROR: Failed to create deployment package"
    exit 1
}
```

```
cd "$ORIGINAL_DIR" || exit 1

if [ ! -f "${TEMP_DIR}/function.zip" ]; then
    echo "ERROR: Deployment package creation failed"
    exit 1
fi

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)

if [ -z "$CREATE_OUTPUT" ]; then
    echo "ERROR: Failed to create Lambda function"
    exit 1
fi

echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
query State --output text" \
    "Active"

################################################################################
# Step 3: Invoke the function
################################################################################

echo ""
echo "============================================"
echo "Step 3: Invoke the function"
echo "============================================"
```

```
echo ""

TEST_EVENT='{"length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

if ! validate_input "$TEST_EVENT" '"length": [0-9]+, "width": [0-9]+'; then
    echo "ERROR: Invalid test event format"
    exit 1
fi

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

if [ ! -f "${TEMP_DIR}/response.json" ]; then
    echo "ERROR: No response file generated"
    exit 1
fi

RESPONSE=$(cat "${TEMP_DIR}/response.json")
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

################################################################################
# Step 4: View CloudWatch logs
################################################################################

echo ""
echo "========================================="
echo "Step 4: View CloudWatch Logs"
echo "========================================="
echo ""
```

```
echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

aws logs tag-log-group \
    --log-group-name "$LOG_GROUP_NAME" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted

################################################################################
```

```
# Summary and cleanup
################################################################################

echo ""
echo "============================================="
echo "SUMMARY"
echo "============================================="
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:   ${FUNCTION_NAME}"
echo "  CloudWatch logs:   ${LOG_GROUP_NAME}"
echo ""
echo "============================================="
echo "CLEANUP"
echo "============================================="
echo ""
echo "Cleaning up all created resources..."
cleanup_resources

echo ""
echo "Done."
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteFunction
  - DeleteLogGroup
  - DeleteRole
  - DescribeLogStreams
  - DetachRolePolicy
  - GetFunction
  - GetFunctionConfiguration
  - GetLogEvents
  - Memohon

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai gudang data tanpa server menggunakan CLI

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Gunakan rahasia-manajer CreateSecret
- Gunakan rahasia-manajer DeleteSecret
- Gunakan rahasia-manajer GetSecretValue
- Gunakan pergeseran merah CreateNamespace
- Gunakan pergeseran merah CreateWorkgroup
- Gunakan pergeseran merah DeleteNamespace
- Gunakan iam CreateRole

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon Redshift Serverless Tutorial Script with Secrets Manager (No jq
 dependency)
# This script creates a Redshift Serverless environment, loads sample data, and
 runs queries
# Uses AWS Secrets Manager for secure password management without requiring jq

# Set up logging
LOG_FILE="redshift-serverless-tutorial-v4.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift Serverless tutorial script at $(date)"
```

```
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
  local output=$1
  local cmd=$2

  if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Command failed: $cmd"
    echo "Output: $output"
    cleanup_resources
    exit 1
  fi
}

# Function to generate a secure password that meets Redshift requirements
generate_secure_password() {
  # Redshift password requirements:
  # - 8-64 characters
  # - At least one uppercase letter
  # - At least one lowercase letter
  # - At least one decimal digit
  # - Can contain printable ASCII characters except /, ", ', \, @, space

  local password=""
  local valid=false
  local attempts=0
  local max_attempts=10

  while [[ "$valid" == false && $attempts -lt $max_attempts ]]; do
    # Generate base password with safe characters
    local base=$(openssl rand -base64 12 | tr -d '/+=' | head -c 12)

    # Ensure we have at least one of each required character type
    local upper=$(echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" | fold -w1 | shuf -n1)
    local lower=$(echo "abcdefghijklmnopqrstuvwxyz" | fold -w1 | shuf -n1)
    local digit=$(echo "0123456789" | fold -w1 | shuf -n1)
    local special=$(echo "!#$%&*()_+-=[]{}|;:,.<>?" | fold -w1 | shuf -n1)

    # Combine and shuffle
    password="${base}${upper}${lower}${digit}${special}"
    password=$(echo "$password" | fold -w1 | shuf | tr -d '\n')

    # Validate password meets requirements
```

```
    if [[ ${#password} -ge 8 && ${#password} -le 64 ]] && \
       [[ "$password" =~ [A-Z] ]] && \
       [[ "$password" =~ [a-z] ]] && \
       [[ "$password" =~ [0-9] ]] && \
       [[ ! "$password" =~ [/\"\'\\@[:space:]] ]]; then
      valid=true
    fi

    ((attempts++))
  done

  if [[ "$valid" == false ]]; then
    echo "ERROR: Failed to generate valid password after $max_attempts attempts"
    exit 1
  fi

  echo "$password"
}

# Function to create secret in AWS Secrets Manager
create_secret() {
  local secret_name=$1
  local username=$2
  local password=$3
  local description=$4

  echo "Creating secret in AWS Secrets Manager: $secret_name"

  # Create the secret using AWS CLI without jq
  local secret_output=$(aws secretsmanager create-secret \
    --name "$secret_name" \
    --description "$description" \
    --secret-string "{\"username\":\"$username\",\"password\":\"$password\"}" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
 2>&1)

  if echo "$secret_output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Failed to create secret: $secret_output"
    return 1
  fi

  echo "Secret created successfully: $secret_name"
  return 0
}
```

```
# Function to retrieve password from AWS Secrets Manager
get_password_from_secret() {
  local secret_name=$1

  # Get the secret value and extract password using sed/grep instead of jq
  local secret_value=$(aws secretsmanager get-secret-value \
    --secret-id "$secret_name" \
    --query 'SecretString' \
    --output text 2>/dev/null)

  if [[ $? -eq 0 ]]; then
    # Extract password from JSON using sed
    echo "$secret_value" | sed -n 's/.*"password":"\([^"]*\)".*/\1/p'
  else
    echo ""
  fi
}

# Function to wait for a resource to be available
wait_for_resource() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be available..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>/dev/null)
    local status=$(echo "$output" | grep -o '"Status": "[^"]*' | cut -d'"' -f4 ||
 echo "")

    if [[ "$status" == "AVAILABLE" ]]; then
      echo "$resource_type $resource_name is now available"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name status:
 $status. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done
```

```
  echo "ERROR: Timed out waiting for $resource_type $resource_name to be
 available"
  return 1
}

# Function to wait for a resource to be deleted
wait_for_resource_deletion() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be deleted..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>&1)

    if echo "$output" | grep -i "not found\|does not exist" > /dev/null; then
      echo "$resource_type $resource_name has been deleted"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name is still being
 deleted. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be deleted"
  return 1
}

# Function to clean up resources
cleanup_resources() {
  echo ""
  echo "========================================="
  echo "CLEANUP CONFIRMATION"
  echo "========================================="
  echo "The following resources were created:"
  echo "- Redshift Serverless Workgroup: $WORKGROUP_NAME"
  echo "- Redshift Serverless Namespace: $NAMESPACE_NAME"
  echo "- IAM Role: $ROLE_NAME"
  echo "- Secrets Manager Secret: $SECRET_NAME"
  echo ""
```

```
  echo "Do you want to clean up all created resources? (y/n): "
  read -r CLEANUP_CHOICE

  if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the workgroup
    echo "Deleting Redshift Serverless workgroup $WORKGROUP_NAME..."
    WORKGROUP_DELETE_OUTPUT=$(aws redshift-serverless delete-workgroup --
workgroup-name "$WORKGROUP_NAME" 2>&1)
    echo "$WORKGROUP_DELETE_OUTPUT"

    # Wait for workgroup to be deleted before deleting namespace
    wait_for_resource_deletion "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-
serverless get-workgroup --workgroup-name $WORKGROUP_NAME"

    # Delete the namespace
    echo "Deleting Redshift Serverless namespace $NAMESPACE_NAME..."
    NAMESPACE_DELETE_OUTPUT=$(aws redshift-serverless delete-namespace --
namespace-name "$NAMESPACE_NAME" 2>&1)
    echo "$NAMESPACE_DELETE_OUTPUT"

    # Wait for namespace to be deleted
    wait_for_resource_deletion "namespace" "$NAMESPACE_NAME" 20 30 "aws redshift-
serverless get-namespace --namespace-name $NAMESPACE_NAME"

    # Delete the IAM role policy
    echo "Deleting IAM role policy..."
    POLICY_DELETE_OUTPUT=$(aws iam delete-role-policy --role-name "$ROLE_NAME" --
policy-name S3Access 2>&1)
    echo "$POLICY_DELETE_OUTPUT"

    # Delete the IAM role
    echo "Deleting IAM role $ROLE_NAME..."
    ROLE_DELETE_OUTPUT=$(aws iam delete-role --role-name "$ROLE_NAME" 2>&1)
    echo "$ROLE_DELETE_OUTPUT"

    # Delete the secret
    echo "Deleting Secrets Manager secret $SECRET_NAME..."
    SECRET_DELETE_OUTPUT=$(aws secretsmanager delete-secret --secret-id
 "$SECRET_NAME" --force-delete-without-recovery 2>&1)
    echo "$SECRET_DELETE_OUTPUT"

    echo "Cleanup completed."
```

```
  else
    echo "Cleanup skipped. Resources will remain in your AWS account."
  fi
}

# Check if required tools are available
if ! command -v openssl &> /dev/null; then
    echo "ERROR: openssl is required but not installed. Please install openssl to
 continue."
    exit 1
fi

# Generate unique names for resources
RANDOM_SUFFIX=$(cat /dev/urandom | tr -dc 'a-z0-9' | head -c 6)
NAMESPACE_NAME="rs-namespace-${RANDOM_SUFFIX}"
WORKGROUP_NAME="rs-workgroup-${RANDOM_SUFFIX}"
ROLE_NAME="RedshiftServerlessS3Role-${RANDOM_SUFFIX}"
SECRET_NAME="redshift-serverless-admin-${RANDOM_SUFFIX}"
DB_NAME="dev"
ADMIN_USERNAME="admin"

# Generate secure password
echo "Generating secure password..."
ADMIN_PASSWORD=$(generate_secure_password)

# Create secret in AWS Secrets Manager
create_secret "$SECRET_NAME" "$ADMIN_USERNAME" "$ADMIN_PASSWORD" "Admin
 credentials for Redshift Serverless namespace $NAMESPACE_NAME"
if [[ $? -ne 0 ]]; then
  echo "ERROR: Failed to create secret in AWS Secrets Manager"
  exit 1
fi

# Track created resources
CREATED_RESOURCES=()

echo "Using the following resource names:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- IAM Role: $ROLE_NAME"
echo "- Secret: $SECRET_NAME"
echo "- Database: $DB_NAME"
echo "- Admin Username: $ADMIN_USERNAME"
echo "- Admin Password: [STORED IN SECRETS MANAGER]"
```

```
# Step 1: Create IAM role for S3 access
echo "Creating IAM role for Redshift Serverless S3 access..."

# Create trust policy document
cat > redshift-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift-serverless.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create S3 access policy document
cat > redshift-s3-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role $ROLE_NAME..."
ROLE_OUTPUT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://redshift-trust-policy.json 2>&1)
```

```
echo "$ROLE_OUTPUT"
check_error "$ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
CREATED_RESOURCES+=("IAM Role: $ROLE_NAME")

# Attach S3 policy to the role
echo "Attaching S3 access policy to role $ROLE_NAME..."
POLICY_OUTPUT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 S3Access --policy-document file://redshift-s3-policy.json 2>&1)
echo "$POLICY_OUTPUT"
check_error "$POLICY_OUTPUT" "aws iam put-role-policy"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Step 2: Create a namespace
echo "Creating Redshift Serverless namespace $NAMESPACE_NAME..."
NAMESPACE_OUTPUT=$(aws redshift-serverless create-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --admin-username "$ADMIN_USERNAME" \
  --admin-user-password "$ADMIN_PASSWORD" \
  --db-name "$DB_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$NAMESPACE_OUTPUT"
check_error "$NAMESPACE_OUTPUT" "aws redshift-serverless create-namespace"
CREATED_RESOURCES+=("Redshift Serverless Namespace: $NAMESPACE_NAME")

# Wait for namespace to be available
wait_for_resource "namespace" "$NAMESPACE_NAME" 10 30 "aws redshift-serverless
 get-namespace --namespace-name $NAMESPACE_NAME"

# Associate IAM role with namespace
echo "Associating IAM role with namespace..."
UPDATE_NAMESPACE_OUTPUT=$(aws redshift-serverless update-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --iam-roles "$ROLE_ARN" 2>&1)
echo "$UPDATE_NAMESPACE_OUTPUT"
check_error "$UPDATE_NAMESPACE_OUTPUT" "aws redshift-serverless update-namespace"

# Step 3: Create a workgroup
echo "Creating Redshift Serverless workgroup $WORKGROUP_NAME..."
```

```
WORKGROUP_OUTPUT=$(aws redshift-serverless create-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --namespace-name "$NAMESPACE_NAME" \
  --base-capacity 8 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$WORKGROUP_OUTPUT"
check_error "$WORKGROUP_OUTPUT" "aws redshift-serverless create-workgroup"
CREATED_RESOURCES+=("Redshift Serverless Workgroup: $WORKGROUP_NAME")

# Wait for workgroup to be available
wait_for_resource "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-serverless
 get-workgroup --workgroup-name $WORKGROUP_NAME"

# Get workgroup endpoint
WORKGROUP_ENDPOINT=$(aws redshift-serverless get-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --query 'workgroup.endpoint.address' \
  --output text)
echo "Workgroup endpoint: $WORKGROUP_ENDPOINT"

# Wait additional time for the endpoint to be fully operational
echo "Waiting for endpoint to be fully operational..."
sleep 60

# Step 4: Create tables for sample data
echo "Creating tables for sample data..."

# Create users table
echo "Creating users table..."
USERS_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE users(
    userid INTEGER NOT NULL DISTKEY SORTKEY,
    username CHAR(8),
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    city VARCHAR(30),
    state CHAR(2),
    email VARCHAR(100),
    phone CHAR(14),
    likesports BOOLEAN,
    liketheatre BOOLEAN,
    likeconcerts BOOLEAN,
```

```
    likejazz BOOLEAN,
    likeclassical BOOLEAN,
    likeopera BOOLEAN,
    likerock BOOLEAN,
    likevegas BOOLEAN,
    likebroadway BOOLEAN,
    likemusicals BOOLEAN
  );" 2>&1)
echo "$USERS_TABLE_OUTPUT"
check_error "$USERS_TABLE_OUTPUT" "aws redshift-data execute-statement (users
 table)"
USERS_QUERY_ID=$(echo "$USERS_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for users table creation to complete..."
sleep 5

# Create event table
echo "Creating event table..."
EVENT_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE event(
    eventid INTEGER NOT NULL DISTKEY,
    venueid SMALLINT NOT NULL,
    catid SMALLINT NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    eventname VARCHAR(200),
    starttime TIMESTAMP
  );" 2>&1)
echo "$EVENT_TABLE_OUTPUT"
check_error "$EVENT_TABLE_OUTPUT" "aws redshift-data execute-statement (event
 table)"
EVENT_QUERY_ID=$(echo "$EVENT_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for event table creation to complete..."
sleep 5

# Create sales table
echo "Creating sales table..."
SALES_TABLE_OUTPUT=$(aws redshift-data execute-statement \
```

```
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE sales(
    salesid INTEGER NOT NULL,
    listid INTEGER NOT NULL DISTKEY,
    sellerid INTEGER NOT NULL,
    buyerid INTEGER NOT NULL,
    eventid INTEGER NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    qtysold SMALLINT NOT NULL,
    pricepaid DECIMAL(8,2),
    commission DECIMAL(8,2),
    saletime TIMESTAMP
  );" 2>&1)
echo "$SALES_TABLE_OUTPUT"
check_error "$SALES_TABLE_OUTPUT" "aws redshift-data execute-statement (sales
 table)"
SALES_QUERY_ID=$(echo "$SALES_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for tables to be created
echo "Waiting for tables to be created..."
sleep 10

# Step 5: Load sample data from Amazon S3
echo "Loading sample data from Amazon S3..."

# Load data into users table
echo "Loading data into users table..."
USERS_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY users
    FROM 's3://redshift-downloads/tickit/allusers_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$USERS_LOAD_OUTPUT"
check_error "$USERS_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 users)"
USERS_LOAD_QUERY_ID=$(echo "$USERS_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)
```

```
# Wait for data loading to complete
echo "Waiting for users data loading to complete..."
sleep 10

# Load data into event table
echo "Loading data into event table..."
EVENT_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY event
    FROM 's3://redshift-downloads/tickit/allevents_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$EVENT_LOAD_OUTPUT"
check_error "$EVENT_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 event)"
EVENT_LOAD_QUERY_ID=$(echo "$EVENT_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for event data loading to complete..."
sleep 10

# Load data into sales table
echo "Loading data into sales table..."
SALES_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY sales
    FROM 's3://redshift-downloads/tickit/sales_tab.txt'
    DELIMITER '\t'
    TIMEFORMAT 'MM/DD/YYYY HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$SALES_LOAD_OUTPUT"
check_error "$SALES_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 sales)"
SALES_LOAD_QUERY_ID=$(echo "$SALES_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for sales data loading to complete..."
```

```
sleep 30

# Step 6: Run sample queries
echo "Running sample queries..."

# Query 1: Find top 10 buyers by quantity
echo "Running query: Find top 10 buyers by quantity..."
QUERY1_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT firstname, lastname, total_quantity
    FROM (SELECT buyerid, sum(qtysold) total_quantity
          FROM sales
          GROUP BY buyerid
          ORDER BY total_quantity desc limit 10) Q, users
    WHERE Q.buyerid = userid
    ORDER BY Q.total_quantity desc;" 2>&1)
echo "$QUERY1_OUTPUT"
check_error "$QUERY1_OUTPUT" "aws redshift-data execute-statement (query 1)"
QUERY1_ID=$(echo "$QUERY1_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 1 to complete..."
sleep 10

# Get query 1 results
echo "Getting results for query 1..."
QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
echo "$QUERY1_STATUS_OUTPUT"
check_error "$QUERY1_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 1)"

QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY1_STATUS" == "FINISHED" ]; then
  QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID" 2>&1)
  echo "Query 1 Results:"
  echo "$QUERY1_RESULTS"
else
  echo "Query 1 is not yet complete. Status: $QUERY1_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20
```

```
  # Check again
  QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
  QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY1_STATUS" == "FINISHED" ]; then
    QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID"
 2>&1)
    echo "Query 1 Results:"
    echo "$QUERY1_RESULTS"
  else
    echo "Query 1 is still not complete. Status: $QUERY1_STATUS"
  fi
fi

# Query 2: Find events in the 99.9 percentile in terms of all time total sales
echo "Running query: Find events in the 99.9 percentile in terms of all time
 total sales..."
QUERY2_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT eventname, total_price
    FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price
 desc) as percentile
          FROM (SELECT eventid, sum(pricepaid) total_price
                FROM sales
                GROUP BY eventid)) Q, event E
    WHERE Q.eventid = E.eventid
    AND percentile = 1
    ORDER BY total_price desc;" 2>&1)
echo "$QUERY2_OUTPUT"
check_error "$QUERY2_OUTPUT" "aws redshift-data execute-statement (query 2)"
QUERY2_ID=$(echo "$QUERY2_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 2 to complete..."
sleep 10

# Get query 2 results
echo "Getting results for query 2..."
QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
echo "$QUERY2_STATUS_OUTPUT"
```

```
check_error "$QUERY2_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 2)"

QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY2_STATUS" == "FINISHED" ]; then
  QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID" 2>&1)
  echo "Query 2 Results:"
  echo "$QUERY2_RESULTS"
else
  echo "Query 2 is not yet complete. Status: $QUERY2_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
  QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
  QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY2_STATUS" == "FINISHED" ]; then
    QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID"
 2>&1)
    echo "Query 2 Results:"
    echo "$QUERY2_RESULTS"
  else
    echo "Query 2 is still not complete. Status: $QUERY2_STATUS"
  fi
fi

# Summary
echo ""
echo "================================================"
echo "TUTORIAL SUMMARY"
echo "================================================"
echo "You have successfully:"
echo "1. Created a Redshift Serverless namespace and workgroup"
echo "2. Created an IAM role with S3 access permissions"
echo "3. Stored admin credentials securely in AWS Secrets Manager"
echo "4. Created tables for sample data"
echo "5. Loaded sample data from Amazon S3"
echo "6. Run sample queries on the data"
echo ""
echo "Redshift Serverless Resources:"
```

```
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- Database: $DB_NAME"
echo "- Endpoint: $WORKGROUP_ENDPOINT"
echo "- Credentials Secret: $SECRET_NAME"
echo ""
echo "To connect to your Redshift Serverless database using SQL tools:"
echo "- Host: $WORKGROUP_ENDPOINT"
echo "- Database: $DB_NAME"
echo "- Username: $ADMIN_USERNAME"
echo "- Password: Retrieve from AWS Secrets Manager secret '$SECRET_NAME'"
echo ""
echo "To retrieve the password from Secrets Manager (without jq):"
echo "aws secretsmanager get-secret-value --secret-id $SECRET_NAME --query
 'SecretString' --output text | sed -n 's/.*\"password\":\"\([^\"]*\)\".*/\1/p'"
echo ""

# Clean up temporary files
rm -f redshift-trust-policy.json redshift-s3-policy.json

# Clean up resources
cleanup_resources

echo "Tutorial completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - CreateNamespace
  - CreateRole
  - CreateSecret
  - CreateWorkgroup
  - DeleteNamespace
  - DeleteRole
  - DeleteRolePolicy
  - DeleteSecret
  - DeleteWorkgroup
  - GetNamespace
  - GetRole

- GetSecretValue
- GetWorkgroup
- PutRolePolicy
- UpdateNamespace

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan manajemen konfigurasi

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat bucket Amazon S3.
- Membuat topik Amazon SNS
- Buat peran IAM untuk Config
- Siapkan perekam konfigurasi Config
- Siapkan saluran pengiriman Config
- Mulai perekam konfigurasi
- Verifikasi pengaturan Config

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI
```

```
# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
```

```
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}
```

```
# Function to display created resources
display_resources() {
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
```

```
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
```

```
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"
```

```
echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15
```

```
# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"
```

```
if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"
```

```
    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"
```

```
echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel

- DeleteRole
- DeleteRolePolicy
- DeleteTopic
- DescribeConfigurationRecorderStatus
- DescribeConfigurationRecorders
- DescribeDeliveryChannels
- DetachRolePolicy
- GetCallerIdentity
- PutConfigurationRecorder
- PutDeliveryChannel
- PutPublicAccessBlock
- PutRolePolicy
- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan perlindungan perangkat internet of things

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat Peran IAM yang Diperlukan
- Aktifkan Pemeriksaan Audit Defender Perangkat IoT
- Jalankan On-Demand Audit
- Buat Aksi Mitigasi
- Terapkan Tindakan Mitigasi pada Temuan
- Mengatur Pemberitahuan SNS (Opsional)
- Aktifkan Logging IoT

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS IoT Device Defender Getting Started Script
# This script demonstrates how to use AWS IoT Device Defender to enable audit
 checks,
# view audit results, create mitigation actions, and apply them to findings.

set -euo pipefail

# Set up logging
LOG_FILE="iot-device-defender-script-$(date +%Y%m%d%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS IoT Device Defender Getting Started Script"
echo "================================================="
echo "Starting script execution at $(date)"
echo ""

# Function to check for errors in command output
check_error() {
    if echo "$1" | grep -iE "An error occurred|Exception|Failed|usage: aws" > /
dev/null; then
        echo "ERROR: Command failed with the following output:"
        echo "$1"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values using jq
extract_json_value() {
```

```
    local json="$1"
    local key="$2"
    echo "$json" | jq -r ".${key} // empty" 2>/dev/null || echo ""
}

# Function to validate JSON
validate_json() {
    local json="$1"
    echo "$json" | jq empty 2>/dev/null
}

# Function to check AWS CLI availability
check_aws_cli() {
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed or not in PATH"
        return 1
    fi
    if ! command -v jq &> /dev/null; then
        echo "ERROR: jq is not installed or not in PATH"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    local account_id
    account_id=$(aws sts get-caller-identity --query 'Account' --output text 2>/
dev/null) || true
    if [ -z "$account_id" ]; then
        echo "ERROR: Could not retrieve AWS account ID"
        return 1
    fi
    echo "$account_id"
    return 0
}

# Function to create IAM roles with retry logic
create_iam_role() {
    local ROLE_NAME=$1
    local TRUST_POLICY=$2
    local MANAGED_POLICY=$3
    local RETRY_COUNT=0
    local MAX_RETRIES=3
```

```
    echo "Creating IAM role: $ROLE_NAME"

    # Validate trust policy JSON
    if ! validate_json "$TRUST_POLICY"; then
        echo "ERROR: Invalid trust policy JSON for role $ROLE_NAME"
        return 1
    fi

    # Check if role already exists
    if aws iam get-role --role-name "$ROLE_NAME" >/dev/null 2>&1; then
        echo "Role $ROLE_NAME already exists, skipping creation"
        ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn'
 --output text 2>/dev/null) || true
        if [ -z "$ROLE_ARN" ]; then
            echo "ERROR: Could not retrieve ARN for existing role $ROLE_NAME"
            return 1
        fi
        echo "Role ARN: $ROLE_ARN"
        return 0
    fi

    # Create the role with trust policy and retry logic
    while [ $RETRY_COUNT -lt $MAX_RETRIES ]; do
        ROLE_RESULT=$(aws iam create-role \
            --role-name "$ROLE_NAME" \
            --assume-role-policy-document "$TRUST_POLICY" 2>&1) || true

        if check_error "$ROLE_RESULT"; then
            break
        fi

        RETRY_COUNT=$((RETRY_COUNT + 1))
        if [ $RETRY_COUNT -lt $MAX_RETRIES ]; then
            echo "Retrying role creation (attempt $((RETRY_COUNT + 1))/
$MAX_RETRIES)..."
            sleep $((RETRY_COUNT * 2))
        fi
    done

    if ! check_error "$ROLE_RESULT"; then
        echo "Failed to create role $ROLE_NAME after $MAX_RETRIES attempts"
        return 1
    fi
```

```
    aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

    # For IoT logging role, create an inline policy instead of using a managed
 policy
    if [[ "$ROLE_NAME" == "AWSIoTLoggingRole" ]]; then
        local LOGGING_POLICY
        LOGGING_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:PutMetricFilter",
                "logs:PutRetentionPolicy",
                "logs:GetLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        }
    ]
}
EOF
)

        if ! validate_json "$LOGGING_POLICY"; then
            echo "ERROR: Invalid logging policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$LOGGING_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
            return 1
        fi
```

```
    elif [[ "$ROLE_NAME" == "IoTMitigationActionErrorLoggingRole" ]]; then
        local MITIGATION_POLICY
        MITIGATION_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:UpdateCACertificate",
                "iot:UpdateCertificate",
                "iot:SetV2LoggingOptions",
                "iot:SetLoggingOptions",
                "iot:AddThingToThingGroup"
            ],
            "Resource": "arn:aws:iot:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "iot.amazonaws.com"
                }
            }
        }
    ]
}
EOF
)

        if ! validate_json "$MITIGATION_POLICY"; then
            echo "ERROR: Invalid mitigation policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$MITIGATION_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
```

```
            return 1
        fi
    else
        # Attach managed policy to role if provided
        if [ -n "$MANAGED_POLICY" ]; then
            ATTACH_RESULT=$(aws iam attach-role-policy \
                --role-name "$ROLE_NAME" \
                --policy-arn "$MANAGED_POLICY" 2>&1) || true

            if ! check_error "$ATTACH_RESULT"; then
                echo "Failed to attach policy to role $ROLE_NAME"
                return 1
            fi
        fi
    fi

    echo "Role $ROLE_NAME created successfully"

    # Get the role ARN with error handling
    ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --
output text 2>/dev/null) || true
    if [ -z "$ROLE_ARN" ]; then
        echo "ERROR: Could not retrieve ARN for newly created role $ROLE_NAME"
        return 1
    fi
    echo "Role ARN: $ROLE_ARN"
    return 0
}

# Array to store created resources for cleanup
declare -a CREATED_RESOURCES

# Validate prerequisites
echo "Validating prerequisites..."
if ! check_aws_cli; then
    echo "ERROR: Prerequisites not met"
    exit 1
fi

ACCOUNT_ID=$(get_account_id) || exit 1
echo "AWS Account ID: $ACCOUNT_ID"
echo ""

# Step 1: Create IAM roles needed for the tutorial
```

```
echo "================================================="
echo "Step 1: Creating required IAM roles"
echo "================================================="

# Create IoT Device Defender Audit role
IOT_DEFENDER_AUDIT_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "AWSIoTDeviceDefenderAuditRole"
 "$IOT_DEFENDER_AUDIT_TRUST_POLICY" "arn:aws:iam::aws:policy/service-role/
AWSIoTDeviceDefenderAudit"; then
    echo "ERROR: Failed to create audit role"
    exit 1
fi
AUDIT_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTDeviceDefenderAuditRole")

# Create IoT Logging role
IOT_LOGGING_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF
)

if ! create_iam_role "AWSIoTLoggingRole" "$IOT_LOGGING_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create logging role"
    exit 1
fi
LOGGING_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTLoggingRole")

# Create IoT Mitigation Action role
IOT_MITIGATION_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "IoTMitigationActionErrorLoggingRole"
 "$IOT_MITIGATION_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create mitigation role"
    exit 1
fi
MITIGATION_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: IoTMitigationActionErrorLoggingRole")

# Wait for IAM role propagation
echo "Waiting for IAM role propagation..."
sleep 5

# Step 2: Enable audit checks
echo ""
echo "================================================="
echo "Step 2: Enabling AWS IoT Device Defender audit checks"
echo "================================================="
```

```
# Get current audit configuration
echo "Getting current audit configuration..."
CURRENT_CONFIG=$(aws iot describe-account-audit-configuration --output json 2>&1)
 || true
if validate_json "$CURRENT_CONFIG"; then
    echo "$CURRENT_CONFIG" | jq '.' 2>/dev/null || echo "Could not parse current
 configuration"
fi

# Enable specific audit checks with proper JSON escaping
echo "Enabling audit checks..."
AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":true}}'

if ! validate_json "$AUDIT_CONFIG"; then
    echo "ERROR: Invalid audit configuration JSON"
    exit 1
fi

UPDATE_RESULT=$(aws iot update-account-audit-configuration \
  --role-arn "$AUDIT_ROLE_ARN" \
  --audit-check-configurations "$AUDIT_CONFIG" 2>&1) || true

if ! check_error "$UPDATE_RESULT"; then
    echo "Failed to update audit configuration"
    exit 1
fi

echo "Audit checks enabled successfully"

# Step 3: Run an on-demand audit
echo ""
echo "================================================="
echo "Step 3: Running an on-demand audit"
echo "================================================="

echo "Starting on-demand audit task..."
AUDIT_TASK_RESULT=$(aws iot start-on-demand-audit-task \
  --target-check-names LOGGING_DISABLED_CHECK --output json 2>&1) || true

if ! check_error "$AUDIT_TASK_RESULT"; then
    echo "Failed to start on-demand audit task"
    exit 1
fi
```

```
TASK_ID=$(extract_json_value "$AUDIT_TASK_RESULT" "taskId")
if [ -z "$TASK_ID" ]; then
    echo "ERROR: Could not extract task ID from response"
    exit 1
fi

echo "Audit task started with ID: $TASK_ID"
CREATED_RESOURCES+=("Audit Task: $TASK_ID")

# Tag the audit task via IoT service
aws iot tag-resource --resource-arn "arn:aws:iot:$(aws configure get region):
${ACCOUNT_ID}:audittask/${TASK_ID}" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

# Wait for the audit task to complete
echo "Waiting for audit task to complete (this may take a few minutes)..."
TASK_STATUS="IN_PROGRESS"
TIMEOUT=0
MAX_TIMEOUT=600
POLL_INTERVAL=15

while [ "$TASK_STATUS" != "COMPLETED" ]; do
    if [ $TIMEOUT -ge $MAX_TIMEOUT ]; then
        echo "WARNING: Audit task did not complete within ${MAX_TIMEOUT} seconds,
 continuing..."
        break
    fi

    sleep "$POLL_INTERVAL"
    TIMEOUT=$((TIMEOUT + POLL_INTERVAL))

    TASK_DETAILS=$(aws iot describe-audit-task --task-id "$TASK_ID" --output json
 2>&1) || true
    if validate_json "$TASK_DETAILS"; then
        TASK_STATUS=$(extract_json_value "$TASK_DETAILS" "taskStatus")
        echo "Current task status: $TASK_STATUS (elapsed: ${TIMEOUT}s)"

        if [ "$TASK_STATUS" = "FAILED" ]; then
            echo "WARNING: Audit task failed, continuing with script..."
            FAILURE_REASON=$(extract_json_value "$TASK_DETAILS"
 "taskStatistics.failedChecksNotApplicable")
            if [ -n "$FAILURE_REASON" ]; then
                echo "Reason: $FAILURE_REASON"
```

```
            fi
            break
        fi
    else
        echo "WARNING: Could not parse task details, retrying..."
    fi
done

echo "Audit task processing completed"

# Get audit findings (non-blocking)
echo "Getting audit findings..."
FINDINGS=$(aws iot list-audit-findings \
  --task-id "$TASK_ID" --output json 2>&1) || true

if validate_json "$FINDINGS"; then
    FINDING_COUNT=$(echo "$FINDINGS" | jq '.findings | length' 2>/dev/null ||
 echo "0")
    echo "Audit findings count: $FINDING_COUNT"
    if [ "$FINDING_COUNT" -gt 0 ]; then
        echo "Sample finding:"
        echo "$FINDINGS" | jq '.findings[0]' 2>/dev/null || echo "Could not parse
 finding"
    fi
else
    echo "WARNING: Could not parse audit findings response"
    FINDINGS='{"findings":[]}'
fi

# Check if we have any non-compliant findings
FINDING_ID=$(extract_json_value "$FINDINGS" "findings[0].findingId")
if [ -n "$FINDING_ID" ]; then
    echo "Found non-compliant finding with ID: $FINDING_ID"
    HAS_FINDINGS=true
else
    echo "No non-compliant findings detected"
    HAS_FINDINGS=false
fi

# Step 4: Create a mitigation action
echo ""
echo "================================================="
echo "Step 4: Creating a mitigation action"
echo "================================================="
```

```
# Check if mitigation action already exists and delete it
if aws iot describe-mitigation-action --action-name "EnableErrorLoggingAction" >/
dev/null 2>&1; then
    echo "Mitigation action 'EnableErrorLoggingAction' already exists, deleting
 it first..."
    aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction"
 2>&1 || true
    sleep 2
fi

echo "Creating mitigation action to enable AWS IoT logging..."

# Build mitigation action parameters JSON
MITIGATION_PARAMS=$(cat <<EOF
{
  "enableIoTLoggingParams": {
    "roleArnForLogging": "$LOGGING_ROLE_ARN",
    "logLevel": "ERROR"
  }
}
EOF
)

if ! validate_json "$MITIGATION_PARAMS"; then
    echo "ERROR: Invalid mitigation parameters JSON"
    exit 1
fi

MITIGATION_RESULT=$(aws iot create-mitigation-action \
  --action-name "EnableErrorLoggingAction" \
  --role-arn "$MITIGATION_ROLE_ARN" \
  --action-params "$MITIGATION_PARAMS" --output json 2>&1) || true

if ! check_error "$MITIGATION_RESULT"; then
    echo "Failed to create mitigation action"
    exit 1
fi

if validate_json "$MITIGATION_RESULT"; then
    echo "Mitigation action created successfully"
    MITIGATION_ACTION_ARN=$(extract_json_value "$MITIGATION_RESULT" "actionArn")
    if [ -n "$MITIGATION_ACTION_ARN" ]; then
        echo "Mitigation Action ARN: $MITIGATION_ACTION_ARN"
```

```
        aws iot tag-resource --resource-arn "$MITIGATION_ACTION_ARN" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs 2>&1
 || true
    fi
else
    echo "WARNING: Could not validate mitigation action response, but action may
 have been created"
fi

CREATED_RESOURCES+=("Mitigation Action: EnableErrorLoggingAction")

# Step 5: Apply mitigation action to findings (if any)
if [ "$HAS_FINDINGS" = true ]; then
    echo ""
    echo "=================================================="
    echo "Step 5: Applying mitigation action to findings"
    echo "=================================================="

    MITIGATION_TASK_ID="MitigationTask-$(date +%s)"
    echo "Starting mitigation actions task with ID: $MITIGATION_TASK_ID"

    # Build target JSON
    TARGET_JSON=$(cat <<EOF
{
  "findingIds": ["$FINDING_ID"]
}
EOF
)

    if ! validate_json "$TARGET_JSON"; then
        echo "ERROR: Invalid target JSON"
        exit 1
    fi

    # Build audit check to actions mapping JSON
    AUDIT_CHECK_MAPPING=$(cat <<EOF
{
  "LOGGING_DISABLED_CHECK": ["EnableErrorLoggingAction"]
}
EOF
)

    if ! validate_json "$AUDIT_CHECK_MAPPING"; then
        echo "ERROR: Invalid audit check mapping JSON"
```

```
        exit 1
    fi

    MITIGATION_TASK_RESULT=$(aws iot start-audit-mitigation-actions-task \
      --task-id "$MITIGATION_TASK_ID" \
      --target "$TARGET_JSON" \
      --audit-check-to-actions-mapping "$AUDIT_CHECK_MAPPING" --output json 2>&1)
 || true

    if ! check_error "$MITIGATION_TASK_RESULT"; then
        echo "WARNING: Failed to start mitigation actions task, continuing..."
    else
        echo "Mitigation actions task started successfully"
        CREATED_RESOURCES+=("Mitigation Task: $MITIGATION_TASK_ID")
    fi
else
    echo ""
    echo "=================================================="
    echo "Step 5: Skipping mitigation action application (no findings)"
    echo "=================================================="
fi

# Step 6: Set up SNS notifications (optional)
echo ""
echo "=================================================="
echo "Step 6: Setting up SNS notifications"
echo "=================================================="

# Check if SNS topic already exists
SNS_TOPICS=$(aws sns list-topics --output json 2>&1) || true
TOPIC_ARN=""
if validate_json "$SNS_TOPICS"; then
    TOPIC_ARN=$(echo "$SNS_TOPICS" | jq -r '.Topics[] | select(.TopicArn |
 contains("IoTDDNotifications")) | .TopicArn' 2>/dev/null | head -1 || echo "")
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "SNS topic 'IoTDDNotifications' already exists, using existing topic..."
    echo "Topic ARN: $TOPIC_ARN"
else
    echo "Creating SNS topic for notifications..."
    SNS_RESULT=$(aws sns create-topic --name "IoTDDNotifications" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs --
output json 2>&1) || true
```

```
    if ! check_error "$SNS_RESULT"; then
        echo "WARNING: Failed to create SNS topic, continuing..."
        SNS_RESULT=""
    else
        TOPIC_ARN=$(extract_json_value "$SNS_RESULT" "TopicArn")
        if [ -n "$TOPIC_ARN" ]; then
            echo "SNS topic created with ARN: $TOPIC_ARN"
            CREATED_RESOURCES+=("SNS Topic: IoTDDNotifications")
        fi
    fi
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "Updating audit configuration to enable SNS notifications..."

    # Build SNS notification configuration JSON
    SNS_CONFIG=$(cat <<EOF
{
  "SNS": {
    "targetArn": "$TOPIC_ARN",
    "roleArn": "$AUDIT_ROLE_ARN",
    "enabled": true
  }
}
EOF
)

    if ! validate_json "$SNS_CONFIG"; then
        echo "ERROR: Invalid SNS configuration JSON"
        exit 1
    fi

    SNS_UPDATE_RESULT=$(aws iot update-account-audit-configuration \
      --audit-notification-target-configurations "$SNS_CONFIG" 2>&1) || true

    if ! check_error "$SNS_UPDATE_RESULT"; then
        echo "WARNING: Failed to update audit configuration for SNS
 notifications"
    else
        echo "SNS notifications enabled successfully"
    fi
else
    echo "Skipping SNS configuration due to topic creation failure"
```

```
fi

# Step 7: Enable AWS IoT logging
echo ""
echo "================================================="
echo "Step 7: Enabling AWS IoT logging"
echo "================================================="

echo "Setting up AWS IoT logging options..."

LOGGING_RESULT=$(aws iot set-v2-logging-options \
  --role-arn "$LOGGING_ROLE_ARN" \
  --default-log-level "ERROR" 2>&1) || true

if ! check_error "$LOGGING_RESULT"; then
    echo "V2 logging setup failed, trying v1 logging..."

    V1_LOGGING_CONFIG=$(cat <<EOF
{
  "roleArn": "$LOGGING_ROLE_ARN",
  "logLevel": "ERROR"
}
EOF
)

    if ! validate_json "$V1_LOGGING_CONFIG"; then
        echo "ERROR: Invalid v1 logging configuration JSON"
        exit 1
    fi

    LOGGING_RESULT_V1=$(aws iot set-logging-options \
      --logging-options-payload "$V1_LOGGING_CONFIG" 2>&1) || true

    if ! check_error "$LOGGING_RESULT_V1"; then
        echo "WARNING: Failed to set up AWS IoT logging with both v1 and v2
 methods, continuing..."
    else
        echo "AWS IoT v1 logging enabled successfully"
    fi
else
    echo "AWS IoT v2 logging enabled successfully"
fi

# Verify logging is enabled
```

```
echo "Verifying logging configuration..."
LOGGING_CONFIG=$(aws iot get-v2-logging-options --output json 2>&1) || true
if [ -n "$LOGGING_CONFIG" ] && ! check_error "$LOGGING_CONFIG" && validate_json
 "$LOGGING_CONFIG"; then
    echo "Logging configuration verified"
    echo "$LOGGING_CONFIG" | jq '.' 2>/dev/null || echo "Configuration retrieved
 but could not display details"
else
    echo "Could not verify logging configuration, but setup completed"
fi

# Script completed successfully
echo ""
echo "================================================="
echo "AWS IoT Device Defender setup completed successfully!"
echo "================================================="
echo "The following resources were created:"
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "- $resource"
done
echo ""

# Cleanup phase
echo "=========================================="
echo "CLEANUP"
echo "=========================================="
echo "Starting automatic cleanup of resources..."
echo "Waiting 10 seconds before cleanup to allow resource stabilization..."
sleep 10

# Disable AWS IoT logging
echo "Disabling AWS IoT logging..."

DISABLE_V2_RESULT=$(aws iot set-v2-logging-options \
  --default-log-level "DISABLED" 2>&1) || true

if check_error "$DISABLE_V2_RESULT"; then
    echo "V2 logging disabled successfully"
else
    echo "Attempting v1 logging disable..."

    V1_DISABLE_CONFIG=$(cat <<'EOF'
{
  "logLevel": "DISABLED"
```

```
}
EOF
)

    DISABLE_V1_RESULT=$(aws iot set-logging-options \
      --logging-options-payload "$V1_DISABLE_CONFIG" 2>&1) || true

    if check_error "$DISABLE_V1_RESULT"; then
        echo "V1 logging disabled successfully"
    else
        echo "WARNING: Could not disable logging"
    fi
fi

# Delete mitigation action
echo "Deleting mitigation action..."
aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction" 2>&1 ||
 true

# Reset audit configuration
echo "Resetting IoT Device Defender audit configuration..."
RESET_AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":false}}'
aws iot update-account-audit-configuration \
  --audit-check-configurations "$RESET_AUDIT_CONFIG" 2>&1 || true

# Delete SNS topic
echo "Deleting SNS topic..."
if [ -n "${TOPIC_ARN:-}" ] && [ "$TOPIC_ARN" != "null" ]; then
    aws sns delete-topic --topic-arn "$TOPIC_ARN" 2>&1 || true
fi

# Clean up IAM roles with improved error handling
echo "Cleaning up IAM roles..."

cleanup_role() {
    local role_name=$1
    echo "Cleaning up role: $role_name"

    if aws iam get-role --role-name "$role_name" >/dev/null 2>&1; then
        ROLE_POLICIES=$(aws iam list-role-policies --role-name "$role_name" --
output json 2>&1 || echo '{"PolicyNames":[]}')
        if validate_json "$ROLE_POLICIES"; then
            while IFS= read -r policy_name; do
                if [ -n "$policy_name" ] && [ "$policy_name" != "null" ]; then
```

```
                    echo "  Deleting inline policy: $policy_name"
                    aws iam delete-role-policy \
                        --role-name "$role_name" \
                        --policy-name "$policy_name" 2>&1 || true
                fi
            done < <(echo "$ROLE_POLICIES" | jq -r '.PolicyNames[]' 2>/dev/null
 || echo "")
        fi

        ATTACHED_POLICIES=$(aws iam list-attached-role-policies --role-name
 "$role_name" --output json 2>&1 || echo '{"AttachedPolicies":[]}')
        if validate_json "$ATTACHED_POLICIES"; then
            while IFS= read -r policy_arn; do
                if [ -n "$policy_arn" ] && [ "$policy_arn" != "null" ]; then
                    echo "  Detaching managed policy: $policy_arn"
                    aws iam detach-role-policy \
                        --role-name "$role_name" \
                        --policy-arn "$policy_arn" 2>&1 || true
                fi
            done < <(echo "$ATTACHED_POLICIES" | jq -r
 '.AttachedPolicies[].PolicyArn' 2>/dev/null || echo "")
        fi

        echo "  Deleting role: $role_name"
        aws iam delete-role --role-name "$role_name" 2>&1 || true
    else
        echo "  Role $role_name does not exist or already deleted"
    fi
}

cleanup_role "AWSIoTDeviceDefenderAuditRole"
cleanup_role "AWSIoTLoggingRole"
cleanup_role "IoTMitigationActionErrorLoggingRole"

echo "Cleanup completed successfully"

echo ""
echo "Script execution completed at $(date)"
echo "Log file: $LOG_FILE"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateMitigationAction

- CreateRole
- CreateTopic
- DeleteAccountAuditConfiguration
- DeleteMitigationAction
- DeleteRole
- DeleteRolePolicy
- DeleteTopic
- DescribeAccountAuditConfiguration
- DescribeAuditTask
- DetachRolePolicy
- GetLoggingOptions
- GetRole
- GetV2LoggingOptions
- ListAuditFindings
- ListAuditMitigationActionsTasks
- ListMitigationActions
- ListRolePolicies
- ListTopics
- PutRolePolicy
- SetLoggingOptions
- SetV2LoggingOptions
- StartAuditMitigationActionsTask
- StartOnDemandAuditTask
- UpdateAccountAuditConfiguration

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan toko fitur pembelajaran mesin

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Siapkan izin IAM
- Buat peran SageMaker eksekusi
- Buat grup fitur
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
```

```
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
```

```
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'
```

```
    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"
```

```
    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

```
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi
```

```
echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
```

```
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
```

```
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
```

```
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
```

```
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)
```

```
if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "==========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "==========================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "==========================================="
echo ""

# Handle cleanup
```

```
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateBucket
  - CreateFeatureGroup
  - CreateRole
  - DeleteBucket

- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan cluster kubernetes terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat VPC untuk kluster EKS Anda
- Buat peran IAM untuk kluster EKS Anda
- Buat kluster EKS Anda
- Konfigurasikan kubectl untuk berkomunikasi dengan klaster Anda
- Membuat grup simpul terkelola
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
```

```
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"
```

```
# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
```

```
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."

# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
```

```
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
```

```
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
```

```
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"

# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
```

```
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"
```

```
NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
```

```
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
```

```
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "========================================="

# Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster

- DeleteNodegroup
- DeleteRole
- DeleteStack
- DescribeCluster
- DescribeNodegroup
- DescribeStacks
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNodegroups
- UpdateKubeconfig
- Tunggu (CloudFormation)
- Tunggu (Amazon EKS)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan streaming terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat kluster MSK
- Buat izin IAM untuk akses MSK
- Buat mesin klien
- Dapatkan broker bootstrap
- Siapkan mesin klien
- Buat topik dan produce/consume data
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
```

```
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi
```

```
    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
```

```
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
    if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
        echo "Terminating EC2 instance: $INSTANCE_ID"
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
 to terminate instance"
        echo "Waiting for instance to terminate..."
        aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
 "Failed to wait for instance termination"
    fi
```

```
   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
       aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
   fi

   # Remove role from instance profile
   if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
       echo "Removing role from instance profile"
       aws iam remove-role-from-instance-profile \
           --instance-profile-name "$INSTANCE_PROFILE_NAME" \
```

```
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
 profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
 "$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
 delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
 "$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
 role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
 delete policy"
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")
```

```
    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
```

```
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
```

```
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
```

```
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi
```

```
if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
```

```
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs
```

```
# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "================================================"
```

```
# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
```

```
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
```

```
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
```

```
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi
```

```
echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
```

```
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
```

```
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
```

```
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
```

```
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
```

```
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "============================================="
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
```

```
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
```

```
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster
  - DeleteInstanceProfile
  - DeleteKeyPair
  - DeletePolicy
  - DeleteRole
  - DeleteSecurityGroup
  - DescribeAvailabilityZones

- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan klaster gudang data yang disediakan

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat cluster Redshift
- Buat peran IAM untuk akses S3
- Buat tabel dan muat data
- Jalankan contoh kueri

- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon Redshift Provisioned Cluster Tutorial Script
# This script creates a Redshift cluster, loads sample data, runs queries, and
 cleans up resources
# Version 4: Security improvements and best practices

set -euo pipefail

# Set up logging
LOG_FILE="redshift_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1" >&2
    echo "Resources created so far:"
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- Redshift Cluster: $CLUSTER_ID"; fi
    if [ -n "${ROLE_NAME:-}" ]; then echo "- IAM Role: $ROLE_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
```

```
    echo "Cleaning up resources..."

    # Delete the cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Deleting Redshift cluster: $CLUSTER_ID"
        aws redshift delete-cluster --cluster-identifier "$CLUSTER_ID" --skip-
final-cluster-snapshot 2>/dev/null || true
        echo "Waiting for cluster deletion to complete..."
        aws redshift wait cluster-deleted --cluster-identifier "$CLUSTER_ID" 2>/
dev/null || true
        echo "Cluster deleted successfully."
    fi

    # Delete the IAM role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Removing IAM role policy..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 RedshiftS3Access 2>/dev/null || true

        echo "Deleting IAM role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    # Clean up temporary files
    rm -f redshift-trust-policy.json redshift-s3-policy.json

    echo "Cleanup completed."
}

# Trap errors and cleanup
trap 'handle_error "Script interrupted"' INT TERM

# Function to wait for SQL statement to complete
wait_for_statement() {
    local statement_id=$1
    local max_attempts=30
    local attempt=1
    local status=""

    echo "Waiting for statement $statement_id to complete..."

    while [ $attempt -le $max_attempts ]; do
        status=$(aws redshift-data describe-statement --id "$statement_id" --
query 'Status' --output text 2>/dev/null || echo "")
```

```
        if [ "$status" == "FINISHED" ]; then
            echo "Statement completed successfully."
            return 0
        elif [ "$status" == "FAILED" ]; then
            local error=$(aws redshift-data describe-statement --id
 "$statement_id" --query 'Error' --output text 2>/dev/null || echo "Unknown
 error")
            echo "Statement failed with error: $error" >&2
            return 1
        elif [ "$status" == "ABORTED" ]; then
            echo "Statement was aborted." >&2
            return 1
        fi

        echo "Statement status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for statement to complete." >&2
    return 1
}

# Function to check if IAM role is attached to cluster
check_role_attached() {
    local role_arn=$1
    local max_attempts=10
    local attempt=1

    echo "Checking if IAM role is attached to the cluster..."

    while [ $attempt -le $max_attempts ]; do
        local status=$(aws redshift describe-clusters \
            --cluster-identifier "$CLUSTER_ID" \
            --query "Clusters[0].IamRoles[?IamRoleArn=='$role_arn'].ApplyStatus"
 \
            --output text 2>/dev/null || echo "")

        if [ "$status" == "in-sync" ]; then
            echo "IAM role is successfully attached to the cluster."
            return 0
        fi
```

```
        echo "IAM role status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 30
        ((attempt++))
    done

    echo "Timed out waiting for IAM role to be attached." >&2
    return 1
}

# Validate required commands
for cmd in aws jq; do
    if ! command -v "$cmd" &> /dev/null; then
        handle_error "Required command '$cmd' not found. Please install it and
 try again."
    fi
done

# Validate AWS credentials
if ! aws sts get-caller-identity &>/dev/null; then
    handle_error "AWS credentials not configured or invalid"
fi

# Variables to track created resources
CLUSTER_ID="examplecluster"
ROLE_NAME="RedshiftS3Role-$(date +%s)"
DB_NAME="dev"
DB_USER="awsuser"

# Generate secure password using AWS Secrets Manager or random string
if command -v openssl &> /dev/null; then
    DB_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-20)
else
    DB_PASSWORD="TempPass$(date +%s | md5sum | cut -c1-20)"
fi

# Validate password meets requirements
if [ ${#DB_PASSWORD} -lt 8 ]; then
    handle_error "Generated password does not meet minimum length requirement"
fi

# Store password securely (optional: use AWS Secrets Manager in production)
echo "Generated database password (store securely): $DB_PASSWORD"
```

```
echo "=== Step 1: Creating Amazon Redshift Cluster ==="

# Create the Redshift cluster with encryption and audit logging enabled
echo "Creating Redshift cluster: $CLUSTER_ID"
CLUSTER_RESULT=$(aws redshift create-cluster \
  --cluster-identifier "$CLUSTER_ID" \
  --node-type ra3.xlplus \
  --number-of-nodes 2 \
  --master-username "$DB_USER" \
  --master-user-password "$DB_PASSWORD" \
  --db-name "$DB_NAME" \
  --port 5439 \
  --encrypted \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-provisioned \
  2>&1) || handle_error "Failed to create Redshift cluster"

echo "$CLUSTER_RESULT"
echo "Waiting for cluster to become available..."

# Wait for the cluster to be available
aws redshift wait cluster-available --cluster-identifier "$CLUSTER_ID" ||
 handle_error "Timeout waiting for cluster to become available"

# Get cluster status to confirm
CLUSTER_STATUS=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].ClusterStatus' \
  --output text)

echo "Cluster status: $CLUSTER_STATUS"

echo "=== Step 2: Creating IAM Role for S3 Access ==="

# Create trust policy file with restricted permissions
echo "Creating trust policy for Redshift"
cat > redshift-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

chmod 600 redshift-trust-policy.json

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://redshift-trust-policy.json 2>&1) ||
 handle_error "Failed to create IAM role"

echo "$ROLE_RESULT"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Tag the IAM role
echo "Tagging IAM role: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=redshift-provisioned

# Create policy document for S3 access with principle of least privilege
echo "Creating S3 access policy"
cat > redshift-s3-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
```

```
    }
  ]
}
EOF

chmod 600 redshift-s3-policy.json

# Attach policy to role
echo "Attaching S3 access policy to role"
POLICY_RESULT=$(aws iam put-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-name RedshiftS3Access \
  --policy-document file://redshift-s3-policy.json 2>&1) || handle_error "Failed
 to attach policy to role"

echo "$POLICY_RESULT"

# Attach role to cluster
echo "Attaching IAM role to Redshift cluster"
ATTACH_ROLE_RESULT=$(aws redshift modify-cluster-iam-roles \
  --cluster-identifier "$CLUSTER_ID" \
  --add-iam-roles "$ROLE_ARN" 2>&1) || handle_error "Failed to attach role to
 cluster"

echo "$ATTACH_ROLE_RESULT"

# Wait for the role to be attached
echo "Waiting for IAM role to be attached to the cluster..."
if ! check_role_attached "$ROLE_ARN"; then
    handle_error "Failed to attach IAM role to cluster"
fi

echo "=== Step 3: Getting Cluster Connection Information ==="

# Get cluster endpoint
CLUSTER_INFO=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].Endpoint.{Address:Address,Port:Port}' \
  --output json)

echo "Cluster endpoint information:"
echo "$CLUSTER_INFO"

echo "=== Step 4: Creating Tables and Loading Data ==="
```

```
echo "Creating sales table"
SALES_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS sales; CREATE TABLE sales(salesid integer not null,
 listid integer not null distkey, sellerid integer not null, buyerid integer
 not null, eventid integer not null, dateid smallint not null sortkey, qtysold
 smallint not null, pricepaid decimal(8,2), commission decimal(8,2), saletime
 timestamp);" \
  --query 'Id' --output text)

echo "Sales table creation statement ID: $SALES_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_TABLE_ID"; then
    handle_error "Failed to create sales table"
fi

echo "Creating date table"
DATE_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS date; CREATE TABLE date(dateid smallint not null
 distkey sortkey, caldate date not null, day character(3) not null, week smallint
 not null, month character(5) not null, qtr character(5) not null, year smallint
 not null, holiday boolean default('N'));" \
  --query 'Id' --output text)

echo "Date table creation statement ID: $DATE_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_TABLE_ID"; then
    handle_error "Failed to create date table"
fi

echo "Loading data into sales table"
SALES_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
```

```
  --sql "COPY sales FROM 's3://redshift-downloads/tickit/sales_tab.txt' DELIMITER
 '\t' TIMEFORMAT 'MM/DD/YYYY HH:MI:SS' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';"
 \
  --query 'Id' --output text)

echo "Sales data load statement ID: $SALES_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_LOAD_ID"; then
    handle_error "Failed to load data into sales table"
fi

echo "Loading data into date table"
DATE_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY date FROM 's3://redshift-downloads/tickit/date2008_pipe.txt'
 DELIMITER '|' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';" \
  --query 'Id' --output text)

echo "Date data load statement ID: $DATE_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_LOAD_ID"; then
    handle_error "Failed to load data into date table"
fi

echo "=== Step 5: Running Example Queries ==="

echo "Running query: Get definition for the sales table"
QUERY1_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT * FROM pg_table_def WHERE tablename = 'sales';" \
  --query 'Id' --output text)

echo "Query 1 statement ID: $QUERY1_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY1_ID"; then
    handle_error "Query 1 failed"
fi
```

```
# Get and display results
echo "Query 1 results (first 10 rows):"
aws redshift-data get-statement-result --id "$QUERY1_ID" --max-items 10

echo "Running query: Find total sales on a given calendar date"
QUERY2_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT sum(qtysold) FROM sales, date WHERE sales.dateid = date.dateid
 AND caldate = '2008-01-05';" \
  --query 'Id' --output text)

echo "Query 2 statement ID: $QUERY2_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY2_ID"; then
    handle_error "Query 2 failed"
fi

# Get and display results
echo "Query 2 results:"
aws redshift-data get-statement-result --id "$QUERY2_ID"

echo "=== Tutorial Complete ==="
echo "The following resources were created:"
echo "- Redshift Cluster: $CLUSTER_ID"
echo "- IAM Role: $ROLE_NAME"

echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Cleaning up all created resources..."
cleanup_resources
echo "All resources have been cleaned up."

# Securely clear password from memory
DB_PASSWORD=""

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .

- CreateCluster
- CreateRole
- DeleteCluster
- DeleteRole
- DeleteRolePolicy
- DescribeClusters
- GetRole
- ModifyClusterIamRoles
- PutRolePolicy
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan orkestrasi alur kerja

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Membuat peran IAM untuk Step Functions
- Buat mesin negara bagian pertama Anda
- Mulai eksekusi mesin negara Anda
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash
```

```
# AWS Step Functions Getting Started Tutorial Script
# This script creates and runs a Step Functions state machine based on the AWS
 Step Functions Getting Started tutorial

set -euo pipefail

# Security: Restrict umask to prevent unintended file permissions
umask 077

# Parse command line arguments
AUTO_CLEANUP=true
while [[ $# -gt 0 ]]; do
    case $1 in
        --auto-cleanup)
            AUTO_CLEANUP=true
            shift
            ;;
        -h|--help)
            echo "Usage: $0 [--auto-cleanup] [--help]"
            echo "  --auto-cleanup: Automatically clean up resources without
 prompting"
            echo "  --help: Show this help message"
            exit 0
            ;;
        *)
            echo "Unknown option: $1"
            echo "Use --help for usage information"
            exit 1
            ;;
    esac
done

# Set up logging with secure permissions
LOG_FILE="step-functions-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"

# Security: Use process substitution with explicit FD cleanup
exec 3>&1 4>&2
exec > >(tee -a "$LOG_FILE") 2>&1
trap 'exec 1>&3 2>&4 3>&- 4>&-' EXIT

echo "Starting AWS Step Functions Getting Started Tutorial..."
```

```
echo "Logging to $LOG_FILE"

# Verify AWS CLI is installed and configured
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS credentials are not configured or invalid"
    exit 1
fi

# Check if jq is available for better JSON parsing
if ! command -v jq &> /dev/null; then
    echo "WARNING: jq is not installed. Using basic JSON parsing which may be
 less reliable."
    echo "Consider installing jq for better error handling: brew install jq
 (macOS) or apt-get install jq (Ubuntu)"
    USE_JQ=false
else
    USE_JQ=true
fi

# Use fixed region that supports Amazon Comprehend
CURRENT_REGION="us-west-2"
echo "Using fixed AWS region: $CURRENT_REGION (supports Amazon Comprehend)"

# Set AWS CLI to use the fixed region for all commands
export AWS_DEFAULT_REGION="$CURRENT_REGION"
export AWS_REGION="$CURRENT_REGION"

# Amazon Comprehend is available in us-west-2, so we can always enable it
echo "Amazon Comprehend is available in region $CURRENT_REGION"
SKIP_COMPREHEND=false

# Security: Initialize all resource variables
STATE_MACHINE_ARN=""
ROLE_NAME=""
ROLE_ARN=""
POLICY_ARN=""
STEPFUNCTIONS_POLICY_ARN=""
EXECUTION_ARN=""
```

```
EXECUTION2_ARN=""
EXECUTION3_ARN=""

# Performance: Cache for AWS API calls to reduce redundant requests
declare -A API_CACHE

# Function to make cached AWS CLI calls
aws_call_cached() {
    local cache_key="$1"
    shift

    if [[ -v API_CACHE["$cache_key"] ]]; then
        echo "${API_CACHE[$cache_key]}"
        return 0
    fi

    local result
    result=$(aws "$@" 2>&1) || return $?
    API_CACHE["$cache_key"]="$result"
    echo "$result"
}

# Function to check for API errors in JSON response with optimized jq usage
check_api_error() {
    local response="$1"
    local operation="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        # Use jq for more reliable JSON parsing with efficient error detection
        if echo "$response" | jq -e '.Error // .error // empty' > /dev/null 2>&1;
 then
            local error_message=$(echo "$response" | jq -r
 '.Error.Message // .Error.Code // .error // "Unknown error"' 2>/dev/null)
            handle_error "$operation failed: $error_message"
        fi
    else
        # Fallback to grep-based detection with optimized pattern
        if echo "$response" | grep -qE '"[Ee]rror":|"error":'; then
            handle_error "$operation failed: $response"
        fi
    fi
}

# Function to extract JSON field efficiently
```

```
extract_json_field() {
    local json="$1"
    local field="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        echo "$json" | jq -r "$field" 2>/dev/null
    else
        echo "$json" | grep -oP "\"${field}\":\s*\"\K[^\"]+|\"${field}\":\s*
\K[^,}]+" | head -1
    fi
}

# Function to securely wait for resource propagation with exponential backoff
wait_for_propagation() {
    local resource_type="$1"
    local wait_time="${2:-10}"

    # Validate wait_time is a positive integer
    if ! [[ "$wait_time" =~ ^[0-9]+$ ]] || [ "$wait_time" -lt 1 ] ||
 [ "$wait_time" -gt 300 ]; then
        echo "WARNING: Invalid wait time $wait_time, using default 10 seconds"
        wait_time=10
    fi

    echo "Waiting for $resource_type to propagate ($wait_time seconds)..."
    sleep "$wait_time"
}

# Function to validate JSON file efficiently
validate_json_file() {
    local file="$1"

    if [[ "$USE_JQ" == "true" ]]; then
        if ! jq empty "$file" 2>/dev/null; then
            handle_error "Invalid JSON in $file"
        fi
    fi
}

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created:"
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
```

```
        echo "- State Machine: $STATE_MACHINE_ARN"
    fi
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "- IAM Role: $ROLE_NAME"
    fi
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "- IAM Policy: $POLICY_ARN"
    fi
    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "- Step Functions Policy: $STEPFUNCTIONS_POLICY_ARN"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to securely clean up resources with parallel deletion
cleanup() {
    echo "Cleaning up resources..."

    # Delete state machine if it exists
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "Deleting state machine: $STATE_MACHINE_ARN"
        aws stepfunctions delete-state-machine --state-machine-arn
 "$STATE_MACHINE_ARN" 2>/dev/null &
    fi

    # Detach and delete policies if they exist
    if [ -n "${POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Comprehend policy $POLICY_ARN from role $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Step Functions policy $STEPFUNCTIONS_POLICY_ARN from role
 $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/null &
    fi

    # Wait for detach operations to complete
    wait 2>/dev/null || true
```

```
    # Delete custom policies if they exist
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "Deleting Comprehend policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "Deleting Step Functions policy: $STEPFUNCTIONS_POLICY_ARN"
        aws iam delete-policy --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/
null &
    fi

    # Wait for policy deletion to complete
    wait 2>/dev/null || true

    # Delete role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || echo "Failed
 to delete role"
    fi

    # Remove temporary files securely
    echo "Removing temporary files"
    local temp_files=(
        "hello-world.json"
        "updated-hello-world.json"
        "sentiment-hello-world.json"
        "step-functions-trust-policy.json"
        "comprehend-policy.json"
        "stepfunctions-policy.json"
        "input.json"
        "sentiment-input.json"
    )

    for file in "${temp_files[@]}"; do
        if [ -f "$file" ]; then
            if command -v shred &> /dev/null; then
                shred -vfz -n 3 "$file" 2>/dev/null || rm -f "$file"
            else
                rm -f "$file"
            fi
        fi
```

```
    done
}

# Security: Set trap to cleanup on script exit
trap cleanup EXIT

# Generate a secure random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
ROLE_NAME="StepFunctionsHelloWorldRole-${RANDOM_ID}"
POLICY_NAME="DetectSentimentPolicy-${RANDOM_ID}"
STATE_MACHINE_NAME="MyFirstStateMachine-${RANDOM_ID}"

echo "Using random identifier: $RANDOM_ID"
echo "Role name: $ROLE_NAME"
echo "Policy name: $POLICY_NAME"
echo "State machine name: $STATE_MACHINE_NAME"

# Step 1: Create the state machine definition
echo "Creating state machine definition..."
cat > hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Result": {
        "IsHelloWorldExample": true,
        "ExecutionWaitTimeInSeconds": 10
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
```

```
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
```

```
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Create IAM role trust policy
echo "Creating IAM role trust policy..."
cat > step-functions-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "states.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://step-functions-trust-policy.json 2>&1)

check_api_error "$ROLE_RESULT" "Create IAM role"
echo "Role created successfully"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-step-functions-gs

# Get the role ARN
ROLE_ARN=$(extract_json_field "$ROLE_RESULT" ".Role.Arn")

if [ -z "$ROLE_ARN" ]; then
    handle_error "Failed to extract role ARN"
fi
```

```
echo "Role ARN: $ROLE_ARN"

# Create a custom policy for Step Functions with least privilege
echo "Creating custom policy for Step Functions..."
cat > stepfunctions-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution",
        "states:DescribeExecution",
        "states:StopExecution"
      ],
      "Resource": "arn:aws:states:*:*:stateMachine:*"
    }
  ]
}
EOF

# Create the policy
echo "Creating Step Functions policy..."
STEPFUNCTIONS_POLICY_RESULT=$(aws iam create-policy \
  --policy-name "StepFunctionsPolicy-${RANDOM_ID}" \
  --policy-document file://stepfunctions-policy.json 2>&1)

check_api_error "$STEPFUNCTIONS_POLICY_RESULT" "Create Step Functions policy"
echo "Step Functions policy created successfully"

# Get the policy ARN
STEPFUNCTIONS_POLICY_ARN=$(extract_json_field "$STEPFUNCTIONS_POLICY_RESULT"
 ".Policy.Arn")

if [ -z "$STEPFUNCTIONS_POLICY_ARN" ]; then
    handle_error "Failed to extract Step Functions policy ARN"
fi
echo "Step Functions policy ARN: $STEPFUNCTIONS_POLICY_ARN"

# Attach policy to the role
echo "Attaching Step Functions policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>&1)
```

```
if [ $? -ne 0 ]; then
    handle_error "Failed to attach Step Functions policy to role: $ATTACH_RESULT"
fi

# Wait for role to propagate (IAM changes can take time to propagate)
wait_for_propagation "IAM role" 8

# Create state machine
echo "Creating state machine: $STATE_MACHINE_NAME"
SM_RESULT=$(aws stepfunctions create-state-machine \
  --name "$STATE_MACHINE_NAME" \
  --definition file://hello-world.json \
  --role-arn "$ROLE_ARN" \
  --type STANDARD \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-step-functions-gs
 2>&1)

check_api_error "$SM_RESULT" "Create state machine"
echo "State machine created successfully"

# Get the state machine ARN
STATE_MACHINE_ARN=$(extract_json_field "$SM_RESULT" ".stateMachineArn")

if [ -z "$STATE_MACHINE_ARN" ]; then
    handle_error "Failed to extract state machine ARN"
fi
echo "State machine ARN: $STATE_MACHINE_ARN"

# Step 2: Start the state machine execution
echo "Starting state machine execution..."
EXEC_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello001-${RANDOM_ID}" 2>&1)

check_api_error "$EXEC_RESULT" "Start execution"
echo "Execution started successfully"

# Get the execution ARN
EXECUTION_ARN=$(extract_json_field "$EXEC_RESULT" ".executionArn")

if [ -z "$EXECUTION_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
```

```
echo "Execution ARN: $EXECUTION_ARN"

# Wait for execution to complete (the workflow has a 10-second wait state)
echo "Waiting for execution to complete (12 seconds)..."
sleep 12

# Check execution status
echo "Checking execution status..."
EXEC_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION_ARN" 2>&1)

echo "Execution status: $EXEC_STATUS"

# Step 3: Update state machine to process external input
echo "Updating state machine to process external input..."
cat > updated-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
```

```
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
```

```
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Update state machine
echo "Updating state machine..."
UPDATE_RESULT=$(aws stepfunctions update-state-machine \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --definition file://updated-hello-world.json \
  --role-arn "$ROLE_ARN" 2>&1)

check_api_error "$UPDATE_RESULT" "Update state machine"
echo "State machine updated successfully"

# Create input file with strict validation
echo "Creating input file..."
cat > input.json << 'EOF'
{
  "wait": 5,
  "hello_world": true
}
EOF

# Validate input JSON
validate_json_file "input.json"

# Start execution with input
echo "Starting execution with input..."
EXEC2_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello002-${RANDOM_ID}" \
  --input file://input.json 2>&1)

check_api_error "$EXEC2_RESULT" "Start execution with input"
echo "Execution with input started successfully"

# Get the execution ARN
EXECUTION2_ARN=$(extract_json_field "$EXEC2_RESULT" ".executionArn")

if [ -z "$EXECUTION2_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
```

```
echo "Execution ARN: $EXECUTION2_ARN"

# Wait for execution to complete (the workflow has a 5-second wait state)
echo "Waiting for execution to complete (8 seconds)..."
sleep 8

# Check execution status
echo "Checking execution status..."
EXEC2_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION2_ARN" 2>&1)

echo "Execution status: $EXEC2_STATUS"

# Step 4: Integrate Amazon Comprehend for sentiment analysis (if available)
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Creating policy for Amazon Comprehend access with least privilege..."
    cat > comprehend-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "comprehend:DetectSentiment"
      ],
      "Resource": "*"
    }
  ]
}
EOF

    # Create policy
    echo "Creating IAM policy: $POLICY_NAME"
    POLICY_RESULT=$(aws iam create-policy \
      --policy-name "$POLICY_NAME" \
      --policy-document file://comprehend-policy.json 2>&1)

    check_api_error "$POLICY_RESULT" "Create Comprehend policy"
    echo "Comprehend policy created successfully"

    # Get policy ARN
    POLICY_ARN=$(extract_json_field "$POLICY_RESULT" ".Policy.Arn")

    if [ -z "$POLICY_ARN" ]; then
```

```
        handle_error "Failed to extract policy ARN"
    fi
    echo "Policy ARN: $POLICY_ARN"

    # Attach policy to role
    echo "Attaching policy to role..."
    ATTACH2_RESULT=$(aws iam attach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN" 2>&1)

    if [ $? -ne 0 ]; then
        handle_error "Failed to attach policy to role: $ATTACH2_RESULT"
    fi

    # Create updated state machine definition with sentiment analysis
    echo "Creating updated state machine definition with sentiment analysis..."
    cat > sentiment-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example with sentiment analysis",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait",
        "FeedbackComment.$": "$.feedback_comment"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "DetectSentiment"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
```

```
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "DetectSentiment": {
      "Type": "Task",
      "Resource": "arn:aws:states:::aws-sdk:comprehend:detectSentiment",
      "Parameters": {
        "LanguageCode": "en",
```

```
        "Text.$": "$.FeedbackComment"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    }
  }
}
EOF

    # Validate sentiment state machine JSON
    validate_json_file "sentiment-hello-world.json"

    # Wait for IAM changes to propagate
    wait_for_propagation "IAM changes" 8

    # Update state machine
    echo "Updating state machine with sentiment analysis..."
    UPDATE2_RESULT=$(aws stepfunctions update-state-machine \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --definition file://sentiment-hello-world.json \
      --role-arn "$ROLE_ARN" 2>&1)

    check_api_error "$UPDATE2_RESULT" "Update state machine with sentiment
 analysis"
    echo "State machine updated with sentiment analysis successfully"

    # Create input file with feedback comment
    echo "Creating input file with feedback comment..."
    cat > sentiment-input.json << 'EOF'
{
  "hello_world": false,
  "wait": 5,
  "feedback_comment": "This getting started with Step Functions workshop is a
 challenge!"
}
EOF

    # Validate sentiment input JSON
    validate_json_file "sentiment-input.json"

    # Start execution with sentiment analysis input
    echo "Starting execution with sentiment analysis input..."
```

```
    EXEC3_RESULT=$(aws stepfunctions start-execution \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --name "hello003-${RANDOM_ID}" \
      --input file://sentiment-input.json 2>&1)

    check_api_error "$EXEC3_RESULT" "Start execution with sentiment analysis"
    echo "Execution with sentiment analysis started successfully"

    # Get the execution ARN
    EXECUTION3_ARN=$(extract_json_field "$EXEC3_RESULT" ".executionArn")

    if [ -z "$EXECUTION3_ARN" ]; then
        handle_error "Failed to extract execution ARN"
    fi
    echo "Execution ARN: $EXECUTION3_ARN"

    # Wait for execution to complete
    echo "Waiting for execution to complete (3 seconds)..."
    sleep 3

    # Check execution status
    echo "Checking execution status..."
    EXEC3_STATUS=$(aws stepfunctions describe-execution \
      --execution-arn "$EXECUTION3_ARN" 2>&1)

    echo "Execution status: $EXEC3_STATUS"
else
    echo "Skipping Amazon Comprehend integration (not available in
 $CURRENT_REGION)"
    EXECUTION3_ARN=""
fi

# Display summary of resources created
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
echo "State Machine: $STATE_MACHINE_ARN"
echo "IAM Role: $ROLE_NAME"
echo "Step Functions Policy: StepFunctionsPolicy-${RANDOM_ID}
 ($STEPFUNCTIONS_POLICY_ARN)"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Comprehend Policy: $POLICY_NAME ($POLICY_ARN)"
fi
```

```
echo "Executions:"
echo "  - hello001-${RANDOM_ID}: $EXECUTION_ARN"
echo "  - hello002-${RANDOM_ID}: $EXECUTION2_ARN"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "  - hello003-${RANDOM_ID}: $EXECUTION3_ARN"
fi
echo "========================================="

# Cleanup
echo ""
echo "========================================="
echo "CLEANUP"
echo "========================================="
echo "Auto-cleanup enabled. Cleaning up resources..."

echo "All resources have been cleaned up."

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateStateMachine
  - DeletePolicy
  - DeleteRole
  - DeleteStateMachine
  - DescribeExecution
  - DetachRolePolicy
  - StartExecution
  - UpdateStateMachine

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola kunci akses IAM menggunakan AWS SDK

Contoh kode berikut menunjukkan cara mengelola kunci akses.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat dan daftar kunci akses.
- Cari tahu kapan dan bagaimana kunci akses terakhir digunakan.
- Perbarui dan hapus kunci akses.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan kunci akses IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
```

```
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
```

```
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise
```

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Gunakan fungsi pembungkus untuk melakukan tindakan kunci akses untuk pengguna saat ini.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
```

```
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola kebijakan IAM menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat dan daftar kebijakan.
- Buat dan dapatkan versi kebijakan.
- Kembalikan kebijakan ke versi sebelumnya.
- Hapus kebijakan.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan kebijakan IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
```

```
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
```

```
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.
```

```
    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
```

```
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Gunakan fungsi pembungkus untuk membuat kebijakan, memperbarui versi, dan mendapatkan informasi tentangnya.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::amzn-s3-demo-bucket"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
```

```
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola peran IAM menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM.
- Lampirkan dan lepaskan kebijakan untuk suatu peran.
- Hapus peran.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan peran IAM.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
```

```
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
```

```
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Gunakan fungsi pembungkus untuk membuat peran, lalu lampirkan dan lepaskan kebijakan.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
 "batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
```

```
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola akun IAM Anda menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Dapatkan dan perbarui alias akun.
- Hasilkan laporan pengguna dan kredensyal.
- Dapatkan ringkasan penggunaan akun.
- Dapatkan detail untuk semua pengguna, grup, peran, dan kebijakan di akun Anda, termasuk hubungan mereka satu sama lain.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat fungsi yang membungkus tindakan akun IAM.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
```

```
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
```

```
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Fungsi pembungkus panggilan untuk mengubah alias akun dan untuk mendapatkan laporan tentang akun.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
```

```
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
```

```
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM
- Buat rahasia di Secrets Manager
- Perbarui kode aplikasi Anda
- Perbarui rahasianya
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash
```

```
# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi
```

```
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
```

```
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)
```

```
# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
```

```
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
```

```
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)
```

```
check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
```

```
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memungkinkan AWS Compute Optimizer Fitur otomatisasi untuk menerapkan tindakan yang disarankan

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memungkinkan fitur AWS Compute Optimizer Otomatisasi menerapkan tindakan yang disarankan

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aco-automation.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk mengaktifkan Otomasi di seluruh organisasi Anda

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memungkinkan Otomasi di seluruh organisasi Anda

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
```

```
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:AssociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:DisassociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:ListAccounts",
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk mengaktifkan Otomasi untuk akun Anda

Contoh kode berikut menunjukkan cara mengaktifkan Kebijakan berbasis izin ini untuk akun Anda

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memberikan akses penuh ke Otomatisasi Pengoptimal Komputasi untuk akun manajemen organisasi

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
               "ec2:DescribeVolumes",
               "organizations:ListAccounts",
               "organizations:DescribeOrganization",
               "organizations:DescribeAccount",
               "organizations:EnableAWSServiceAccess",
               "organizations:ListDelegatedAdministrators",
               "organizations:RegisterDelegatedAdministrator",
               "organizations:DeregisterDelegatedAdministrator"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk memberikan akses penuh ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memberikan akses penuh ke Otomatisasi Pengoptimal Komputasi untuk akun mandiri AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
            "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk akun manajemen organisasi

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memberikan akses hanya-baca ke Otomatisasi Pengoptimal Komputasi untuk akun manajemen organisasi

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
               "aco-automation:GetAutomationEvent",
               "aco-automation:GetAutomationRule",
               "aco-automation:ListAccounts",
               "aco-automation:ListAutomationEvents",
```

```
                "aco-automation:ListAutomationEventSteps",
                "aco-automation:ListAutomationEventSummaries",
                "aco-automation:ListAutomationRules",
                "aco-automation:ListAutomationRulePreview",
                "aco-automation:ListAutomationRulePreviewSummaries",
                "aco-automation:ListRecommendedActions",
                "aco-automation:ListRecommendedActionSummaries",
                "aco-automation:ListTagsForResource",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk memberikan akses hanya-baca ke Otomatisasi Compute Optimizer untuk mandiri AWS rekening

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memberikan akses hanya-baca ke Otomatisasi Pengoptimal Komputasi untuk akun mandiri AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aco-automation:GetEnrollmentConfiguration",
                "aco-automation:GetAutomationEvent",
                "aco-automation:GetAutomationRule",
                "aco-automation:ListAutomationEvents",
                "aco-automation:ListAutomationEventSteps",
                "aco-automation:ListAutomationEventSummaries",
                "aco-automation:ListAutomationRules",
                "aco-automation:ListAutomationRulePreview",
```

```
                "aco-automation:ListAutomationRulePreviewSummaries",
                "aco-automation:ListRecommendedActions",
                "aco-automation:ListRecommendedActionSummaries",
                "aco-automation:ListTagsForResource",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kebijakan untuk memberikan izin peran terkait layanan untuk Otomatisasi Pengoptimalan Komputasi

Contoh kode berikut menunjukkan cara Kebijakan berbasis izin ini memberikan izin peran terkait layanan untuk Otomatisasi Pengoptimalan Komputasi

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": "iam:PutRolePolicy",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        }
    ]
```

```
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Kembalikan versi kebijakan IAM menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Dapatkan daftar versi kebijakan secara berurutan berdasarkan tanggal.
- Temukan versi kebijakan default.
- Jadikan versi kebijakan sebelumnya sebagai default.
- Hapus versi default yang lama.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
```

```
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Untuk detail API, lihat topik berikut ini adalah Referensi API SDK untuk Python (Boto3)AWS .
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM
- Buat CloudWatch alarm
- Buat template eksperimen
- Jalankan percobaan
- Verifikasi hasilnya
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial
```

```
#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
```

```
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
```

```
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
```

```
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
```

```
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
```

```
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"
```

```
    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
```

```
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
```

```
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done
```

```
if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
```

```
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "=========================================="
echo "RESOURCES CREATED"
echo "=========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "=========================================="

# Prompt for cleanup
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
```

```
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
"$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
"$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
```

```
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy

- GetCallerIdentity
- GetExperiment
- GetMetricData
- GetRole
- MonitorInstances
- PutMetricAlarm
- PutRolePolicy
- RemoveRoleFromInstanceProfile
- RunInstances
- StartExperiment
- StopExperiment
- TerminateInstances
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengatur Attribute-Based Access Control untuk DynamoDB menggunakan AWS Command Line Interface v2

Contoh kode berikut menunjukkan bagaimana menerapkan Attribute-Based Access Control (ABAC) untuk DynamoDB.

- Buat kebijakan IAM untuk ABAC.
- Buat tabel dengan tag untuk departemen yang berbeda.
- Daftar dan filter tabel berdasarkan tag.

Bash

AWS CLI dengan skrip Bash

Buat kebijakan IAM untuk ABAC.

```
# Step 1: Create a policy document for ABAC
```

```
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

Buat tabel dengan tag untuk departemen yang berbeda.

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
        Key=Environment,Value=Development

# Create another table with different tags
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

Daftar dan filter tabel berdasarkan tag.

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text |
 xargs -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn"
 --output text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
```

```
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - CreatePolicy
  - CreateTable
  - ListTables

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Menyiapkan manajemen sistem

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat izin IAM untuk Systems Manager
- Membuat peran IAM untuk Systems Manager
- Konfigurasikan Systems Manager
- Verifikasi pengaturan
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
```

```
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"
```

```
        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""
```

```
# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
```

```
        ],
        "Resource": "*"
      },
      {
        "Sid": "SsmDocument",
        "Effect": "Allow",
        "Action": [
          "ssm:GetDocument",
          "ssm:DescribeDocument"
        ],
        "Resource": [
          "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
          "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
        ]
      },
      {
        "Sid": "SsmEnableExplorer",
        "Effect": "Allow",
        "Action": "ssm:StartAutomationExecution",
        "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
      },
      {
        "Sid": "SsmExplorerRds",
        "Effect": "Allow",
        "Action": [
          "ssm:GetOpsSummary",
          "ssm:CreateResourceDataSync",
          "ssm:UpdateResourceDataSync"
        ],
        "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
      },
      {
        "Sid": "OrgsReadOnly",
        "Effect": "Allow",
        "Action": [
          "organizations:DescribeAccount",
          "organizations:DescribeOrganization",
          "organizations:ListDelegatedAdministrators",
          "organizations:ListRoots",
          "organizations:ListParents",
          "organizations:ListOrganizationalUnitsForParent",
          "organizations:DescribeOrganizationalUnit",
          "organizations:ListAWSServiceAccessForOrganization"
        ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
            "resource-explorer-2.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "CfnReadOnly",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackSets",
        "cloudformation:DescribeOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
```

```
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
    {
      "Sid": "CfnStackSetActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
```

```
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
```

```
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
```

```
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
```

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
```

```
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF
```

```
echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "=========================================="
```

```
echo "CREATED RESOURCES"
echo "========================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy

- GetCallerIdentity
- GetConfigurationManager

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Menggunakan variabel properti dalam memantau dasbor untuk memantau beberapa fungsi tanpa server

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat fungsi Lambda untuk pemantauan
- Buat CloudWatch dasbor
- Tambahkan variabel properti ke dasbor
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# CloudWatch Dashboard with Lambda Function Variable Script
# This script creates a CloudWatch dashboard with a property variable for Lambda
 function names

set -euo pipefail

# Security: Set restrictive umask
umask 0077
```

```
# Set up logging with secure permissions
LOG_FILE="cloudwatch-dashboard-script-v4.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
echo "Starting script execution at $(date)" >> "$LOG_FILE"

# Function to log commands and their output (with sensitive data sanitization)
log_cmd() {
    local cmd="$1"
    local sanitized_cmd="${cmd//--password*/--password [REDACTED]}"
    sanitized_cmd="${sanitized_cmd//--secret*/--secret [REDACTED]}"
    echo "$(date): Running command: $sanitized_cmd" >> "$LOG_FILE"
    eval "$cmd" 2>&1 | tee -a "$LOG_FILE"
    return ${PIPESTATUS[0]}
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [ $cmd_status -ne 0 ] || echo "$cmd_output" | grep -qi "error"; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        # Sanitize output before logging
        local sanitized_output="${cmd_output//arn:aws:iam::[0-9]*/
arn:aws:iam::ACCOUNT_ID}"
        echo "Command output: $sanitized_output" | tee -a "$LOG_FILE"
        cleanup_resources
        exit 1
    fi
}

# Trap errors and cleanup
trap 'cleanup_resources' EXIT ERR INT TERM

# Function to clean up resources
cleanup_resources() {
    local exit_code=$?

    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
```

```
    if [ -n "${DASHBOARD_NAME:-}" ]; then
        echo "Deleting CloudWatch dashboard: $DASHBOARD_NAME" | tee -a
 "$LOG_FILE"
        aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" 2>&1
 >> "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION1:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION1" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION2:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION2" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole 2>&1 >>
 "$LOG_FILE" || true

        echo "Deleting IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam delete-role --role-name "$ROLE_NAME" 2>&1 >> "$LOG_FILE" || true
    fi

    # Clean up temporary files securely
    shred -vfz -n 3 trust-policy.json lambda_function.py lambda_function.zip 2>/
dev/null || rm -f trust-policy.json lambda_function.py lambda_function.zip

    echo "Cleanup completed." | tee -a "$LOG_FILE"

    return $exit_code
}

# Validate AWS CLI is installed and authenticated
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed" | tee -a "$LOG_FILE"
    exit 1
fi
```

```
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not properly authenticated" | tee -a "$LOG_FILE"
    exit 1
fi

# Get AWS region with validation
AWS_REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION" | tee -a
 "$LOG_FILE"
else
    echo "Using AWS region: $AWS_REGION" | tee -a "$LOG_FILE"
fi

# Validate region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    echo "ERROR: Invalid AWS region format: $AWS_REGION" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate unique identifiers using secure random with validation
RANDOM_ID=$(openssl rand -hex 6)
if [ -z "$RANDOM_ID" ] || [ ${#RANDOM_ID} -ne 12 ]; then
    echo "ERROR: Failed to generate valid random identifier" | tee -a "$LOG_FILE"
    exit 1
fi

DASHBOARD_NAME="LambdaMetricsDashboard-${RANDOM_ID}"
LAMBDA_FUNCTION1="TestFunction1-${RANDOM_ID}"
LAMBDA_FUNCTION2="TestFunction2-${RANDOM_ID}"
ROLE_NAME="LambdaExecutionRole-${RANDOM_ID}"

# Validate resource names don't exceed AWS limits
if [ ${#DASHBOARD_NAME} -gt 128 ] || [ ${#LAMBDA_FUNCTION1} -gt 64 ] ||
 [ ${#ROLE_NAME} -gt 64 ]; then
    echo "ERROR: Generated resource names exceed AWS limits" | tee -a "$LOG_FILE"
    exit 1
fi

echo "Using random identifier: $RANDOM_ID" | tee -a "$LOG_FILE"
echo "Dashboard name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
```

```
echo "Lambda function names: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "IAM role name: $ROLE_NAME" | tee -a "$LOG_FILE"

# Create IAM role for Lambda functions
echo "Creating IAM role for Lambda..." | tee -a "$LOG_FILE"
TRUST_POLICY='{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}'

echo "$TRUST_POLICY" > trust-policy.json
chmod 600 trust-policy.json

# Validate JSON before use
if ! python3 -m json.tool trust-policy.json > /dev/null 2>&1; then
    echo "ERROR: Invalid trust policy JSON" | tee -a "$LOG_FILE"
    exit 1
fi

ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name '$ROLE_NAME' --assume-
role-policy-document file://trust-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=cloudwatch-streams --output json")
check_error "$ROLE_OUTPUT" $? "Failed to create IAM role"

ROLE_ARN=$(echo "$ROLE_OUTPUT" | python3 -c "import sys, json;
 print(json.load(sys.stdin)['Role']['Arn'])" 2>/dev/null)
if [ -z "$ROLE_ARN" ]; then
    echo "ERROR: Failed to extract Role ARN" | tee -a "$LOG_FILE"
    exit 1
fi
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Attach Lambda basic execution policy to the role
echo "Attaching Lambda execution policy to role..." | tee -a "$LOG_FILE"
```

```
POLICY_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name '$ROLE_NAME' --
policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole")
check_error "$POLICY_OUTPUT" $? "Failed to attach policy to role"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..." | tee -a "$LOG_FILE"
sleep 10

# Create simple Python Lambda function code with security validation
echo "Creating Lambda function code..." | tee -a "$LOG_FILE"
cat > lambda_function.py << 'LAMBDA_EOF'
def handler(event, context):
    print("Lambda function executed successfully")
    return {
        'statusCode': 200,
        'body': 'Success'
    }
LAMBDA_EOF

chmod 600 lambda_function.py

# Validate Python syntax
if ! python3 -m py_compile lambda_function.py 2>/dev/null; then
    echo "ERROR: Invalid Python syntax in Lambda function" | tee -a "$LOG_FILE"
    exit 1
fi

# Zip the Lambda function code
zip -j -q lambda_function.zip lambda_function.py
if [ ! -f lambda_function.zip ]; then
    echo "ERROR: Failed to create lambda_function.zip" | tee -a "$LOG_FILE"
    exit 1
fi
chmod 600 lambda_function.zip

# Validate zip file integrity
if ! unzip -t lambda_function.zip > /dev/null 2>&1; then
    echo "ERROR: Created zip file is corrupted" | tee -a "$LOG_FILE"
    exit 1
fi

# Create first Lambda function
echo "Creating first Lambda function: $LAMBDA_FUNCTION1..." | tee -a "$LOG_FILE"
```

```
LAMBDA1_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION1' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA1_OUTPUT" $? "Failed to create first Lambda function"

# Create second Lambda function
echo "Creating second Lambda function: $LAMBDA_FUNCTION2..." | tee -a "$LOG_FILE"
LAMBDA2_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION2' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA2_OUTPUT" $? "Failed to create second Lambda function"

# Invoke Lambda functions to generate some metrics
echo "Invoking Lambda functions to generate metrics..." | tee -a "$LOG_FILE"
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION1' --payload '{}' /
dev/null" || true
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION2' --payload '{}' /
dev/null" || true

# Create CloudWatch dashboard with property variable
echo "Creating CloudWatch dashboard with property variable..." | tee -a
 "$LOG_FILE"

# Create dashboard body with proper escaping and validation
DASHBOARD_BODY=$(cat <<EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "$LAMBDA_FUNCTION1" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$AWS_REGION",
        "title": "Lambda Invocations",
        "period": 300,
```

```
        "stat": "Sum"
      }
    }
  ]
}
EOF
)

# Validate JSON before sending
if ! echo "$DASHBOARD_BODY" | python3 -m json.tool > /dev/null 2>&1; then
    echo "ERROR: Dashboard body is not valid JSON" | tee -a "$LOG_FILE"
    exit 1
fi

# First create a basic dashboard without variables
echo "Creating initial dashboard without variables..." | tee -a "$LOG_FILE"
DASHBOARD_OUTPUT=$(aws cloudwatch put-dashboard --dashboard-name
 "$DASHBOARD_NAME" --dashboard-body "$DASHBOARD_BODY" --output json 2>&1)
check_error "$DASHBOARD_OUTPUT" $? "Failed to create initial CloudWatch
 dashboard"

# Now let's try to add a property variable using the console instructions
echo "To complete the tutorial, please follow these steps in the CloudWatch
 console:" | tee -a "$LOG_FILE"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/" | tee -a "$LOG_FILE"
echo "2. Navigate to Dashboards and select your dashboard: $DASHBOARD_NAME" | tee
 -a "$LOG_FILE"
echo "3. Choose Actions > Variables > Create a variable" | tee -a "$LOG_FILE"
echo "4. Choose Property variable" | tee -a "$LOG_FILE"
echo "5. For Property that the variable changes, choose FunctionName" | tee -a
 "$LOG_FILE"
echo "6. For Input type, choose Select menu (dropdown)" | tee -a "$LOG_FILE"
echo "7. Choose Use the results of a metric search" | tee -a "$LOG_FILE"
echo "8. Choose Pre-built queries > Lambda > Errors" | tee -a "$LOG_FILE"
echo "9. Choose By Function Name and then choose Search" | tee -a "$LOG_FILE"
echo "10. (Optional) Configure any secondary settings as desired" | tee -a
 "$LOG_FILE"
echo "11. Choose Add variable" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "The dashboard has been created and can be accessed at:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"
```

```
# Verify dashboard creation
echo "Verifying dashboard creation..." | tee -a "$LOG_FILE"
VERIFY_OUTPUT=$(aws cloudwatch get-dashboard --dashboard-name "$DASHBOARD_NAME"
 --output json 2>&1)
check_error "$VERIFY_OUTPUT" $? "Failed to verify dashboard creation"

echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "DASHBOARD CREATED SUCCESSFULLY" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "Dashboard Name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda Functions: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "You can view your dashboard in the CloudWatch console:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Auto-confirm cleanup
echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "The following resources were created:" | tee -a "$LOG_FILE"
echo "- CloudWatch Dashboard: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
echo "- IAM Role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "Auto-confirming cleanup of all created resources..." | tee -a "$LOG_FILE"

echo "Script completed successfully." | tee -a "$LOG_FILE"
exit 0
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards

- DeleteFunction
- DeleteRole
- DetachRolePolicy
- GetDashboard
- Memohon
- PutDashboard

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Bekerja dengan DynamoDB Streams dan menggunakan Time-to-Live AWS Command Line Interface v2

Contoh kode berikut menunjukkan cara mengelola DynamoDB Streams dan fitur. Time-to-Live

- Buat tabel dengan Streams diaktifkan.
- Jelaskan Aliran.
- Buat fungsi Lambda untuk memproses Streams.
- Aktifkan TTL di atas meja.
- Tambahkan item dengan atribut TTL.
- Jelaskan pengaturan TTL.

Bash

AWS CLI dengan skrip Bash

Buat tabel dengan Streams diaktifkan.

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
```

```
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

Jelaskan Aliran.

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"

# Describe the stream
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

Buat fungsi Lambda untuk Streams.

```
# Step 1: Create an IAM role for the Lambda function
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
```

```
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

Aktifkan TTL di atas meja.

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo

# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

Tambahkan item dengan atribut TTL.

```
# Calculate expiration time (current time + 1 day in seconds)
```

```
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

Jelaskan pengaturan TTL.

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
    --table-name TTLDemo
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateRole
  - CreateTable
  - DescribeTable
  - DescribeTimeToLive
  - PutItem
  - UpdateTimeToLive

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Bekerja dengan API Pembuat Kebijakan IAM menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat kebijakan IAM dengan menggunakan API berorientasi objek.
- Gunakan API Pembuat Kebijakan IAM dengan layanan IAM.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Contoh menggunakan impor berikut.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
```

```
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Buat kebijakan berbasis waktu.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_GREATER_THAN)

 .key("aws:CurrentTime")

 .value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_LESS_THAN)

 .key("aws:CurrentTime")

 .value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
 readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Buat kebijakan dengan beberapa kondisi.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
```

```
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

 .addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

 .addAction("dynamodb:BatchWriteItem")

 .addResource("arn:aws:dynamodb:*:*:table/table-name")

 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                    List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")

 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Gunakan prinsipal dalam kebijakan.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
```

```
                                                .addPrincipal(IamPrincipal.ALL)
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Izinkan akses lintas akun.

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                        .key("s3:x-amz-
acl")
                                                                        .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Bangun dan unggah file`IamPolicy`.

```
       public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
               // Build the policy.
               IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                               .addStatement(IamStatement.builder()
                                               .effect(IamEffect.ALLOW)
                                               .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
                                                                       + ":table/
exampleTableName")
                                               .build())
                               .build();
               // Upload the policy.
               iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
               return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
       }
```

Unduh dan bekerja dengan file`IamPolicy`.

```
       public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
                       String newPolicyName) {

               String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
 policyName;
               GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
 r.policyArn(policyArn));

               String policyVersion =
 getPolicyResponse.policy().defaultVersionId();
               GetPolicyVersionResponse getPolicyVersionResponse = iam
                               .getPolicyVersion(r ->
 r.policyArn(policyArn).versionId(policyVersion));

               // Create an IamPolicy instance from the JSON string returned
 from IAM.
               String decodedPolicy =
 URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
```

```
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS SDK for Java 2.x.
- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode untuk AWS STS memakai AWS SDK

Contoh kode berikut menunjukkan cara menggunakan AWS STS kit pengembangan AWS perangkat lunak (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

AWS STS

- Contoh dasar untuk AWS STS memakai AWS SDK
  - Tindakan untuk AWS STS memakai AWS SDK
    - Gunakan AssumeRoledengan AWS SDK atau CLI
    - Gunakan AssumeRoleWithWebIdentitydengan CLI
    - Gunakan DecodeAuthorizationMessagedengan CLI
    - Gunakan GetFederationTokendengan CLI
    - Gunakan GetSessionTokendengan AWS SDK atau CLI
- Skenario untuk AWS STS memakai AWS SDK
  - Asumsikan peran IAM yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK
  - Konfigurasikan konektivitas layanan kontainer
  - Membangun URL dengan AWS STS untuk pengguna federasi yang menggunakan AWS SDK
  - Buat tugas kontainer untuk tipe peluncuran tanpa server
  - Buat API istirahat dengan integrasi proxy fungsi
  - Membuat layanan kontainer untuk instance mesin virtual
  - Membuat ruang kerja pemantauan terkelola
  - Membuat dasbor pemantauan dengan nama fungsi sebagai variabel

- Dapatkan token sesi yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK
- Memulai dengan manajemen konfigurasi
- Memulai dengan pendaftar kontainer
- Memulai dengan pesan internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan pemberitahuan push
- Memulai dengan mesin pencari dan analitik
- Memulai dengan firewall aplikasi web
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem

Contoh dasar untuk AWS STS memakai AWS SDK

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar AWS Security Token Service dengan AWS SDK.

Contoh

- Tindakan untuk AWS STS memakai AWS SDK
  - Gunakan AssumeRoledengan AWS SDK atau CLI
  - Gunakan AssumeRoleWithWebIdentitydengan CLI
  - Gunakan DecodeAuthorizationMessagedengan CLI
  - Gunakan GetFederationTokendengan CLI
  - Gunakan GetSessionTokendengan AWS SDK atau CLI

Tindakan untuk AWS STS memakai AWS SDK

Contoh kode berikut menunjukkan cara melakukan AWS STS tindakan individual dengan AWS SDK. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Kutipan ini memanggil AWS STS API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Anda dapat melihat tindakan dalam konteks diSkenario untuk AWS STS memakai AWS SDK.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi AWS Security Token Service API.

Contoh

- Gunakan AssumeRoledengan AWS SDK atau CLI
- Gunakan AssumeRoleWithWebIdentitydengan CLI
- Gunakan DecodeAuthorizationMessagedengan CLI
- Gunakan GetFederationTokendengan CLI
- Gunakan GetSessionTokendengan AWS SDK atau CLI

Gunakan **`AssumeRole`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AssumeRole`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Asumsikan peran IAM yang membutuhkan token MFA
- Membangun URL untuk pengguna federasi

.NET

SDK untuk .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
```

```
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
```

```
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
```

```
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }
```

```
  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }
```

```
    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk C++ API.

CLI

AWS CLI

Untuk mengambil peran

`assume-role`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran IAM. `s3-access-example`

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Output:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

Output dari perintah berisi kunci akses, kunci rahasia, dan token sesi yang dapat Anda gunakan untuk AWS mengautentikasi.

Untuk penggunaan AWS CLI, Anda dapat mengatur profil bernama yang terkait dengan peran. Saat Anda menggunakan profil, AWS CLI akan memanggil peran assume-dan mengelola kredensil untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan peran IAM di AWS CLI di Panduan Pengguna AWS CLI.

- Untuk detail API, lihat AssumeRoledi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
```

```
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();
```

```
        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat klien.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asumsikan peran IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk JavaScript API.

SDK untuk JavaScript (v2)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});
```

```
//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk JavaScript API.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensil yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari MFA yang terkait dengan kredensil pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk detail API, lihat AssumeRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan satu set kredensi sementara (kunci akses, kunci rahasia, dan token sesi) yang dapat digunakan selama satu jam untuk mengakses AWS sumber daya yang biasanya tidak dapat diakses oleh pengguna yang meminta. Kredensil yang dikembalikan memiliki izin yang diizinkan oleh kebijakan akses dari peran yang diasumsikan dan kebijakan yang diberikan (Anda tidak dapat menggunakan kebijakan yang disediakan untuk memberikan izin melebihi yang ditentukan oleh kebijakan akses peran yang diasumsikan).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Contoh 2: Mengembalikan satu set kredensi sementara, berlaku selama satu jam, yang memiliki izin yang sama yang ditentukan dalam kebijakan akses peran yang diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Contoh 3: Mengembalikan satu set kredensi sementara yang memasok nomor seri dan token yang dihasilkan dari MFA yang terkait dengan kredensil pengguna yang digunakan untuk mengeksekusi cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Contoh 4: Mengembalikan satu set kredensi sementara yang telah mengambil peran yang ditentukan dalam akun pelanggan. Untuk setiap peran yang dapat diasumsikan oleh pihak ketiga, akun pelanggan harus membuat peran menggunakan pengidentifikasi yang harus diteruskan dalam ExternalId parameter - setiap kali peran diasumsikan.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Untuk detail API, lihat AssumeRoledi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Asumsikan peran IAM yang memerlukan token MFA dan gunakan kredensil sementara untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
```

```
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk detail API, lihat AssumeRoledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Untuk detail API, lihat AssumeRoledi Referensi AWS SDK untuk Ruby API.

Rust

SDK for Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Untuk detail API, lihat AssumeRolereferensi AWS SDK for Rust API.

Swift

SDK para Swift

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

```
import AWSSTS
```

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
    async throws -> STSClientTypes.Credentials
{
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        print("Error assuming role: ", dump(error))
        throw error
    }
}
```

- Untuk detail API, lihat referensi AssumeRole AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AssumeRoleWithWebIdentity`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Untuk mendapatkan kredensi jangka pendek untuk peran yang diautentikasi dengan Identitas Web (OAuth 2."0)

`assume-role-with-web-identity`Perintah berikut mengambil satu set kredensi jangka pendek untuk peran IAM. `app1` Permintaan diautentikasi dengan menggunakan token

identitas web yang disediakan oleh penyedia identitas web yang ditentukan. Dua kebijakan tambahan diterapkan pada sesi untuk lebih membatasi apa yang dapat dilakukan pengguna. Kredensi yang dikembalikan kedaluwarsa satu jam setelah dibuat.

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Output:

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A",
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    },
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Untuk informasi lebih lanjut, lihat Meminta Kredensial Keamanan Sementara dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat AssumeRoleWithWebIdentitydi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan kumpulan kredensional sementara, berlaku selama satu jam, untuk pengguna yang telah diautentikasi dengan penyedia identitas Login with Amazon. Kredensi mengasumsikan kebijakan akses yang terkait dengan peran yang diidentifikasi oleh peran ARN. Secara opsional, Anda dapat meneruskan kebijakan JSON ke parameter -Policy yang selanjutnya menyempurnakan izin akses (Anda tidak dapat memberikan izin lebih dari yang tersedia di izin yang terkait dengan peran tersebut). Nilai yang diberikan ke -WebIdentityToken adalah pengidentifikasi pengguna unik yang dikembalikan oleh penyedia identitas.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Untuk detail API, lihat AssumeRoleWithWebIdentitydi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan kumpulan kredensional sementara, berlaku selama satu jam, untuk pengguna yang telah diautentikasi dengan penyedia identitas Login with Amazon. Kredensi mengasumsikan kebijakan akses yang terkait dengan peran yang diidentifikasi oleh peran ARN. Secara opsional, Anda dapat meneruskan kebijakan JSON ke parameter -Policy yang selanjutnya menyempurnakan izin akses (Anda tidak dapat memberikan izin lebih dari yang tersedia di izin yang terkait dengan peran tersebut). Nilai yang diberikan ke -WebIdentityToken adalah pengidentifikasi pengguna unik yang dikembalikan oleh penyedia identitas.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Untuk detail API, lihat AssumeRoleWithWebIdentitydi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DecodeAuthorizationMessage`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`DecodeAuthorizationMessage`.

CLI

AWS CLI

Untuk memecahkan kode pesan otorisasi yang dikodekan dikembalikan sebagai tanggapan atas permintaan

`decode-authorization-message`Contoh berikut menerjemahkan informasi tambahan tentang status otorisasi permintaan dari pesan yang disandikan yang dikembalikan sebagai tanggapan atas permintaan Amazon Web Services.

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Output:

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
```

```
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```

Untuk informasi selengkapnya, lihat Logika evaluasi kebijakan di Panduan Pengguna AWS IAM.

- Untuk detail API, lihat DecodeAuthorizationMessagedi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mendekode informasi tambahan yang terkandung dalam konten pesan disandikan yang disediakan yang dikembalikan sebagai tanggapan atas permintaan. Informasi tambahan dikodekan karena rincian status otorisasi dapat merupakan informasi istimewa yang tidak boleh dilihat oleh pengguna yang meminta tindakan.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Untuk detail API, lihat DecodeAuthorizationMessagedi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mendekode informasi tambahan yang terkandung dalam konten pesan disandikan yang disediakan yang dikembalikan sebagai tanggapan atas permintaan. Informasi tambahan

dikodekan karena rincian status otorisasi dapat merupakan informasi istimewa yang tidak boleh dilihat oleh pengguna yang meminta tindakan.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Untuk detail API, lihat DecodeAuthorizationMessagedi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetFederationToken`**dengan CLI

Contoh kode berikut menunjukkan cara menggunakan`GetFederationToken`.

CLI

AWS CLI

Untuk mengembalikan satu set kredensi keamanan sementara menggunakan kredensi kunci akses pengguna IAM

`get-federation-token`Contoh berikut mengembalikan satu set kredensi keamanan sementara (terdiri dari ID kunci akses, kunci akses rahasia, dan token keamanan) untuk pengguna. Anda harus memanggil `GetFederationToken` operasi menggunakan kredensi keamanan jangka panjang dari pengguna IAM.

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Isi dari `myfile.json`:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
```

```
    "PackedPolicySize": 36
}
```

Untuk informasi lebih lanjut, lihat Meminta Kredensial Keamanan Sementara dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetFederationTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Meminta token federasi yang valid selama satu jam menggunakan “Bob” sebagai nama pengguna federasi. Nama ini dapat digunakan untuk mereferensikan nama pengguna federasi dalam kebijakan berbasis sumber daya (seperti kebijakan bucket Amazon S3). Kebijakan IAM yang disediakan, dalam format JSON, digunakan untuk mencakup izin yang tersedia untuk pengguna IAM. Kebijakan yang diberikan tidak dapat memberikan izin lebih dari yang diberikan kepada pengguna yang meminta, dengan izin akhir untuk pengguna federasi menjadi set yang paling ketat berdasarkan persimpangan kebijakan yang disahkan dan kebijakan pengguna IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Untuk detail API, lihat GetFederationTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Meminta token federasi yang valid selama satu jam menggunakan “Bob” sebagai nama pengguna federasi. Nama ini dapat digunakan untuk mereferensikan nama pengguna federasi dalam kebijakan berbasis sumber daya (seperti kebijakan bucket Amazon S3). Kebijakan IAM yang disediakan, dalam format JSON, digunakan untuk mencakup izin yang tersedia untuk pengguna IAM. Kebijakan yang diberikan tidak dapat memberikan izin lebih dari yang diberikan kepada pengguna yang meminta, dengan izin akhir untuk pengguna federasi menjadi set yang paling ketat berdasarkan persimpangan kebijakan yang disahkan dan kebijakan pengguna IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Untuk detail API, lihat GetFederationTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetSessionToken`**dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetSessionToken`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Dapatkan token sesi yang membutuhkan token MFA

CLI

AWS CLI

Untuk mendapatkan satu set kredensi jangka pendek untuk identitas IAM

`get-session-token`Perintah berikut mengambil satu set kredensi jangka pendek untuk identitas IAM yang membuat panggilan. Kredensi yang dihasilkan dapat digunakan untuk permintaan di mana otentikasi multi-faktor (MFA) diperlukan oleh kebijakan. Kredensialnya kedaluwarsa 15 menit setelah dibuat.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Output:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
```

```
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Untuk informasi lebih lanjut, lihat Meminta Kredensial Keamanan Sementara dalam AWS Panduan Pengguna IAM.

- Untuk detail API, lihat GetSessionTokendi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell V4

Contoh 1: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

```
AccessKeyId                          Expiration
 SecretAccessKey                     SessionToken
-----------                          ----------
 ---------------                     ------------
EXAMPLEACCESSKEYID                   2/16/2015 9:12:28 PM
 examplesecretaccesskey...           SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensi sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                          Expiration
 SecretAccessKey                     SessionToken
```

```
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Contoh 3: Mengembalikan **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensi sementara yang valid selama satu jam menggunakan nomor identifikasi perangkat MFA yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang diberikan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Untuk detail API, lihat GetSessionTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V4).

Alat untuk PowerShell V5

Contoh 1: Mengembalikan sebuah **`Amazon.RuntimeAWSCredentials`** instance yang berisi kredensial sementara yang valid untuk jangka waktu tertentu. Kredensial yang digunakan untuk meminta kredensil sementara disimpulkan dari default shell saat ini. Untuk menentukan kredensil lainnya, gunakan parameter - ProfileName atau - AccessKey SecretKey /-.

```
Get-STSSessionToken
```

Output:

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

```
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

Contoh 2: Mengembalikan sebuah **Amazon.RuntimeAWSCredentials** instance yang berisi kredensi sementara yang valid selama satu jam. Kredensi yang digunakan untuk membuat permintaan diperoleh dari profil yang ditentukan.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Output:

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

Contoh 3: Mengembalikan **Amazon.RuntimeAWSCredentials** instance yang berisi kredensi sementara yang valid selama satu jam menggunakan nomor identifikasi perangkat MFA yang terkait dengan akun yang kredensialnya ditentukan dalam profil 'myprofilename' dan nilai yang diberikan oleh perangkat.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Output:

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

- Untuk detail API, lihat GetSessionTokendi Referensi Alat AWS untuk PowerShell Cmdlet (V5).

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Dapatkan token sesi dengan meneruskan token MFA dan gunakan untuk mencantumkan bucket Amazon S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
```

```
    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Untuk detail API, lihat GetSessionTokendi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Skenario untuk AWS STS memakai AWS SDK

Contoh kode berikut menunjukkan cara menerapkan skenario umum AWS STS dengan AWS SDK. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi di dalam AWS STS atau dikombinasikan dengan yang lain Layanan AWS. Setiap skenario menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode.

Skenario menargetkan tingkat pengalaman menengah untuk membantu Anda memahami tindakan layanan dalam konteks.

Contoh

- Asumsikan peran IAM yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK
- Konfigurasikan konektivitas layanan kontainer
- Membangun URL dengan AWS STS untuk pengguna federasi yang menggunakan AWS SDK
- Buat tugas kontainer untuk tipe peluncuran tanpa server
- Buat API istirahat dengan integrasi proxy fungsi
- Membuat layanan kontainer untuk instance mesin virtual
- Membuat ruang kerja pemantauan terkelola
- Membuat dasbor pemantauan dengan nama fungsi sebagai variabel
- Dapatkan token sesi yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK

- Memulai dengan manajemen konfigurasi
- Memulai dengan pendaftar kontainer
- Memulai dengan pesan internet of things
- Memulai dengan toko fitur pembelajaran mesin
- Memulai dengan cluster kubernetes terkelola
- Memulai dengan streaming terkelola
- Memulai dengan pemberitahuan push
- Memulai dengan mesin pencari dan analitik
- Memulai dengan firewall aplikasi web
- Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia
- Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan
- Menyiapkan manajemen sistem

Asumsikan peran IAM yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK

Contoh kode berikut menunjukkan cara mengambil peran yang membutuhkan token MFA.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat peran IAM yang memberikan izin untuk mencantumkan bucket Amazon S3.
- Buat pengguna IAM yang memiliki izin untuk mengambil peran hanya ketika kredensil MFA disediakan.
- Daftarkan perangkat MFA untuk pengguna.
- Asumsikan peran dan gunakan kredensil sementara untuk membuat daftar bucket S3.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM, daftarkan perangkat MFA, dan buat peran yang memberikan izin untuk membuat daftar bucket S3. Pengguna hanya memiliki hak untuk mengambil peran.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
 role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
```

```
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
```

```
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
```

```
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Tunjukkan bahwa mengasumsikan peran tanpa token MFA tidak diperbolehkan.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Asumsikan peran yang memberikan izin untuk mencantumkan bucket S3, meneruskan token MFA yang diperlukan, dan tunjukkan bahwa bucket dapat dicantumkan.

```
def list_buckets_from_assumed_role_with_mfa(
```

```
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
```

```
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Untuk detail API, lihat AssumeRoledi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Konfigurasikan konektivitas layanan kontainer

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat infrastruktur VPC
- Mengatur pencatatan
- Buat cluster ECS
- Konfigurasikan peran IAM
- Buat layanan dengan Service Connect
- Verifikasi penyebaran
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
```

```
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"
```

```
    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"
```

```
    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true
```

```
    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
```

```
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
```

```
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."
```

```
    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"
```

```
    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
```

```
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"
```

```
    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi
```

```
    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "============================================="
    echo "CREATED RESOURCES"
    echo "============================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "============================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
```

```
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
```

```
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups
```

```
    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "============================================"
    echo "CLEANUP CONFIRMATION"
    echo "============================================"
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
```

```
main "$@"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition
  - DescribeClusters
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNamespaces
  - ListServices
  - ListTaskDefinitions
  - ListTasks

- RegisterTaskDefinition
- UpdateService
- Tunggu (Amazon EC2)
- Tunggu (Amazon ECS)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membangun URL dengan AWS STS untuk pengguna federasi yang menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM yang memberikan akses hanya-baca ke sumber daya Amazon S3 akun saat ini.
- Dapatkan token keamanan dari titik akhir AWS federasi.
- Buat URL yang dapat digunakan untuk mengakses konsol dengan kredenal federasi.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat peran yang memberikan akses hanya-baca ke sumber daya S3 akun saat ini.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
```

```
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Dapatkan token keamanan dari titik akhir AWS federasi dan buat URL yang dapat digunakan untuk mengakses konsol dengan kredensi federasi.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
```

```
       AWS federation endpoint.
   3. Builds a URL that can be used in a browser to navigate to the AWS
federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

   :param assume_role_arn: The role that specifies the permissions that are
granted.
                           The current user must have permission to assume the
role.
   :param session_name: The name for the STS session.
   :param issuer: The organization that issues the URL.
   :param sts_client: A Boto3 STS instance that can assume the role.
   :return: The federated URL.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn, RoleSessionName=session_name
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   session_data = {
       "sessionId": temp_credentials["AccessKeyId"],
       "sessionKey": temp_credentials["SecretAccessKey"],
       "sessionToken": temp_credentials["SessionToken"],
   }
   aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

   # Make a request to the AWS federation endpoint to get a sign-in token.
   # The requests.get function URL-encodes the parameters and builds the query
string
   # before making the request.
   response = requests.get(
       aws_federated_signin_endpoint,
       params={
           "Action": "getSigninToken",
           "SessionDuration": str(datetime.timedelta(hours=12).seconds),
           "Session": json.dumps(session_data),
       },
   )
   signin_token = json.loads(response.text)
   print(f"Got a sign-in token from the AWS sign-in federation endpoint.")
```

```
    # Make a federated URL that can be used to sign into the AWS Management
 Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
```

```
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Untuk detail API, lihat AssumeRoledi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Buat tugas kontainer untuk tipe peluncuran tanpa server

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat cluster
- Buat definisi tugas
- Buat layanan
- Bersihkan

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"
```

```
    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "========================================="
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "========================================="

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
```

```
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
```

```
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
```

```
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "==========================================="
    echo "CLEANUP PROCESS"
    echo "==========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."

    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup process..."

        # Step 1: Scale service to 0 tasks first, then delete service
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
 then
            echo ""
            echo "Step 1: Scaling service to 0 tasks..."
            if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
                echo "Waiting for service to stabilize after scaling to 0..."
                execute_command "aws ecs wait services-stable --cluster
 '$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

                echo "Deleting service..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME'" "Delete ECS service"
            else
```

```
                echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
            fi
        fi

        # Step 2: Wait a bit for tasks to fully terminate
        echo ""
        echo "Step 2: Waiting for tasks to fully terminate..."
        sleep 15

        # Step 3: Delete cluster
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
" ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
```

```
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}
```

```
# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "============================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "============================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

```
    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================"
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================"

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================"
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================"

# Create task definition JSON with validated inputs
```

```
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
```

```
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "==========================================="
echo "STEP 4: SET UP NETWORKING"
echo "==========================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
```

```
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
```

```
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "============================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "============================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "============================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "============================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"
```

```
# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "========================================="
```

```
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices
  - DescribeSubnets

- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Buat API istirahat dengan integrasi proxy fungsi

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk eksekusi Lambda
- Buat dan terapkan fungsi Lambda
- Buat REST API
- Konfigurasikan integrasi proxy Lambda
- Menerapkan dan menguji API
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
```

```
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
```

```
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration
```

```
# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
```

```
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
```

```
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
```

```
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddPermission
  - AttachRolePolicy

- CreateDeployment
- CreateFunction
- CreateResource
- CreateRestApi
- CreateRole
- DeleteFunction
- DeleteRestApi
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetResources
- GetRestApis
- PutIntegration
- PutMethod

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat layanan kontainer untuk instance mesin virtual

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat cluster ECS
- Buat dan pantau layanan
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()
```

```
# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi
```

```
        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
        if ! aws ecs deregister-task-definition --task-definition
 "$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to deregister task definition
 $TASK_DEFINITION_ARN"
            ((cleanup_errors++))
        fi
    fi

    # Terminate EC2 instance
    if [[ -n "${INSTANCE_ID:-}" ]]; then
        log "Terminating EC2 instance: $INSTANCE_ID"
        if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
 2>>"$LOG_FILE"; then
            log "WARNING: Failed to terminate instance $INSTANCE_ID"
            ((cleanup_errors++))
```

```
        else
            log "Waiting for instance to terminate..."
            if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
                log "WARNING: Instance termination wait failed for $INSTANCE_ID"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Delete security group with retry logic
    if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
        log "Deleting security group: $SECURITY_GROUP_ID"
        local retry_count=0
        local max_retries=3

        while [[ $retry_count -lt $max_retries ]]; do
            if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
                log "Successfully deleted security group"
                break
            else
                ((retry_count++))
                if [[ $retry_count -lt $max_retries ]]; then
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
$max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
```

```
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"
```

```
    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
```

```
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
```

```
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
```

```
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."
```

```
    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
```

```
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
```

```
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi
```

```
    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"
```

```
    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
```

```
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "============================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "============================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
```

```
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
```

```
main "$@"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeContainerInstances
  - DescribeInstances
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetParameters
  - GetRole
  - ListContainerInstances
  - ListServices
  - ListTasks
  - RegisterTaskDefinition
  - RunInstances

- StopTask
- TerminateInstances
- UpdateService
- Tunggu (Amazon EC2)
- Tunggu (Amazon ECS)
- Tunggu (IAM)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat ruang kerja pemantauan terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk ruang kerja Anda
- Buat ruang kerja Grafana
- Konfigurasikan otentikasi
- Konfigurasikan pengaturan opsional
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it
```

```
# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi
```

```
    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
```

```
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"
```

```
ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."
```

```
    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
```

```
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "========================================="

# Step 6: Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done
```

```
    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace

- DescribeWorkspace
- DetachRolePolicy
- GetCallerIdentity

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat dasbor pemantauan dengan nama fungsi sebagai variabel

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat CloudWatch dasbor
- Tambahkan widget metrik Lambda dengan variabel nama fungsi
- Verifikasi dasbor
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "=========================================="
    echo "CLEANUP CONFIRMATION"
    echo "=========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true

    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi
```

```
# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."

    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi
```

```
    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
```

```
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
```

```
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
```

```
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
```

```
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
```

```
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"
```

```
if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi
```

```
    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetDashboard
  - Memohon
  - ListDashboards
  - ListFunctions
  - PutDashboard

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Dapatkan token sesi yang membutuhkan token MFA dengan AWS STS menggunakan sebuah AWS SDK

Contoh kode berikut menunjukkan cara mendapatkan token sesi yang membutuhkan token MFA.

> **Warning**
>
> Untuk menghindari risiko keamanan, jangan gunakan pengguna IAM untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti AWS IAM Identity Center.

- Buat peran IAM yang memberikan izin untuk mencantumkan bucket Amazon S3.
- Buat pengguna IAM yang memiliki izin untuk mengambil peran hanya ketika kredensil MFA disediakan.
- Daftarkan perangkat MFA untuk pengguna.
- Berikan kredensil MFA untuk mendapatkan token sesi dan gunakan kredensil sementara untuk membuat daftar bucket S3.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di Repositori Contoh Kode AWS.

Buat pengguna IAM, daftarkan perangkat MFA, dan buat peran yang memberikan izin untuk membiarkan pengguna mencantumkan bucket S3 hanya jika kredenal MFA digunakan.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.
```

```
    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
```

```
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
 True}},
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Dapatkan kredenal sesi sementara dengan meneruskan token MFA, dan gunakan kredensialnya untuk mencantumkan bucket S3 untuk akun tersebut.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.
```

```
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Hancurkan sumber daya yang dibuat untuk demo.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
```

```
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Jalankan skenario ini dengan menggunakan fungsi yang ditentukan sebelumnya.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Untuk detail API, lihat GetSessionTokendi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan manajemen konfigurasi

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat bucket Amazon S3.
- Membuat topik Amazon SNS
- Buat peran IAM untuk Config
- Siapkan perekam konfigurasi Config
- Siapkan saluran pengiriman Config
- Mulai perekam konfigurasi
- Verifikasi pengaturan Config

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
```

```
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi
```

```
    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
```

```
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
```

```
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"
```

```
# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"
```

```
echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
```

```
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
```

```
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"
```

```
echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeConfigurationRecorderStatus

- DescribeConfigurationRecorders
- DescribeDeliveryChannels
- DetachRolePolicy
- GetCallerIdentity
- PutConfigurationRecorder
- PutDeliveryChannel
- PutPublicAccessBlock
- PutRolePolicy
- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan pendaftar kontainer

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Membuat citra Docker
- Buat repositori Amazon ECR
- Hapus sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash
```

```
# Amazon ECR Getting Started Script
# This script demonstrates the lifecycle of a Docker image in Amazon ECR

# Set up logging
LOG_FILE="ecr-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "Amazon ECR Getting Started Tutorial"
echo "================================================="
echo "This script will:"
echo "1. Create a Docker image"
echo "2. Create an Amazon ECR repository"
echo "3. Authenticate to Amazon ECR"
echo "4. Push the image to Amazon ECR"
echo "5. Pull the image from Amazon ECR"
echo "6. Clean up resources (optional)"
echo "================================================="

# Check prerequisites
echo "Checking prerequisites..."

# Check if AWS CLI is installed
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed. Please install it before running this
 script."
    echo "Visit https://docs.aws.amazon.com/cli/latest/userguide/install-
cliv2.html for installation instructions."
    exit 1
fi

# Check if AWS CLI is configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not configured properly. Please run 'aws configure'
 to set up your credentials."
    exit 1
fi

# Check if Docker is installed
if ! command -v docker &> /dev/null; then
    echo "ERROR: Docker is not installed. Please install Docker before running
 this script."
```

```
    echo "Visit https://docs.docker.com/get-docker/ for installation
 instructions."
    exit 1
fi

# Check if Docker daemon is running
if ! docker info &> /dev/null; then
    echo "ERROR: Docker daemon is not running. Please start Docker and try
 again."
    exit 1
fi

echo "All prerequisites met."

# Initialize variables
REPO_URI=""
TIMEOUT_CMD="timeout 300"  # 5-minute timeout for long-running commands

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Check the log file for details: $LOG_FILE"

    echo "=================================================="
    echo "Resources created:"
    echo "- Docker image: hello-world (local)"
    if [ -n "$REPO_URI" ]; then
        echo "- ECR Repository: hello-repository"
        echo "- ECR Image: $REPO_URI:latest"
    fi
    echo "=================================================="

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "=================================================="
    echo "Cleaning up resources..."

    # Delete the image from ECR if it exists
    if [ -n "$REPO_URI" ]; then
```

```
        echo "Deleting image from ECR repository..."
        aws ecr batch-delete-image --repository-name hello-repository --image-ids
 imageTag=latest || echo "Failed to delete image, it may not exist or may have
 already been deleted."
    fi

    # Delete the ECR repository if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting ECR repository..."
        aws ecr delete-repository --repository-name hello-repository --force ||
 echo "Failed to delete repository, it may not exist or may have already been
 deleted."
    fi

    # Remove local Docker image
    echo "Removing local Docker image..."
    docker rmi hello-world:latest 2>/dev/null || echo "Failed to remove local
 image, it may not exist or may have already been deleted."
    if [ -n "$REPO_URI" ]; then
        docker rmi "$REPO_URI:latest" 2>/dev/null || echo "Failed to remove
 tagged image, it may not exist or may have already been deleted."
    fi

    echo "Cleanup completed."
    echo "================================================="
}

# Step 1: Create a Docker image
echo "Step 1: Creating a Docker image"

# Create Dockerfile
echo "Creating Dockerfile..."
cat > Dockerfile << 'EOF'
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Install dependencies
RUN yum update -y && \
 yum install -y httpd

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
```

```
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
EOF

# Build Docker image
echo "Building Docker image..."
$TIMEOUT_CMD docker build -t hello-world . || handle_error "Failed to build
 Docker image or operation timed out after 5 minutes"

# Verify image was created
echo "Verifying Docker image..."
docker images --filter reference=hello-world || handle_error "Failed to list
 Docker images"

echo "Docker image created successfully."

# Step 2: Create an Amazon ECR repository
echo "Step 2: Creating an Amazon ECR repository"

# Get AWS account ID
echo "Getting AWS account ID..."
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ -z "$AWS_ACCOUNT_ID" || "$AWS_ACCOUNT_ID" == *"error"* ]]; then
    handle_error "Failed to get AWS account ID. Make sure your AWS credentials
 are configured correctly."
fi
echo "AWS Account ID: $AWS_ACCOUNT_ID"

# Get current region
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
    echo "No AWS region configured, defaulting to $AWS_REGION"
else
    echo "Using AWS region: $AWS_REGION"
fi

# Create ECR repository
echo "Creating ECR repository..."
```

```
REPO_RESULT=$(aws ecr create-repository --repository-name hello-repository --
tags key=project,value=doc-smith key=tutorial,value=amazon-elastic-container-
registry-gs)
if [[ -z "$REPO_RESULT" || "$REPO_RESULT" == *"error"* ]]; then
    handle_error "Failed to create ECR repository"
fi

# Extract repository URI
REPO_URI="$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/hello-repository"
echo "Repository URI: $REPO_URI"

# Step 3: Authenticate to Amazon ECR
echo "Step 3: Authenticating to Amazon ECR"

echo "Getting ECR login password..."
aws ecr get-login-password --region "$AWS_REGION" | docker login --username
 AWS --password-stdin "$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com" ||
 handle_error "Failed to authenticate to ECR"

echo "Successfully authenticated to ECR."

# Step 4: Push the image to Amazon ECR
echo "Step 4: Pushing the image to Amazon ECR"

# Tag the image
echo "Tagging Docker image..."
docker tag hello-world:latest "$REPO_URI:latest" || handle_error "Failed to tag
 Docker image"

# Push the image with timeout
echo "Pushing image to ECR..."
$TIMEOUT_CMD docker push "$REPO_URI:latest" || handle_error "Failed to push image
 to ECR or operation timed out after 5 minutes"

echo "Successfully pushed image to ECR."

# Step 5: Pull the image from Amazon ECR
echo "Step 5: Pulling the image from Amazon ECR"

# Remove local image to ensure we're pulling from ECR
echo "Removing local tagged image..."
docker rmi "$REPO_URI:latest" || echo "Warning: Failed to remove local tagged
 image"
```

```
# Pull the image with timeout
echo "Pulling image from ECR..."
$TIMEOUT_CMD docker pull "$REPO_URI:latest" || handle_error "Failed to pull image
 from ECR or operation timed out after 5 minutes"

echo "Successfully pulled image from ECR."

# List resources created
echo "================================================="
echo "Resources created:"
echo "- Docker image: hello-world (local)"
echo "- ECR Repository: hello-repository"
echo "- ECR Image: $REPO_URI:latest"
echo "================================================="

# Ask user if they want to clean up resources
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    # Step 6: Delete the image from ECR
    echo "Step 6: Deleting the image from ECR"

    DELETE_IMAGE_RESULT=$(aws ecr batch-delete-image --repository-name hello-
repository --image-ids imageTag=latest)
    if [[ -z "$DELETE_IMAGE_RESULT" || "$DELETE_IMAGE_RESULT" == *"error"* ]];
 then
        echo "Warning: Failed to delete image from ECR"
    else
        echo "Successfully deleted image from ECR."
    fi

    # Step 7: Delete the ECR repository
    echo "Step 7: Deleting the ECR repository"

    DELETE_REPO_RESULT=$(aws ecr delete-repository --repository-name hello-
repository --force)
    if [[ -z "$DELETE_REPO_RESULT" || "$DELETE_REPO_RESULT" == *"error"* ]]; then
        echo "Warning: Failed to delete ECR repository"
    else
```

```
        echo "Successfully deleted ECR repository."
    fi

    # Remove local Docker images
    echo "Removing local Docker images..."
    docker rmi hello-world:latest 2>/dev/null || echo "Warning: Failed to remove
 local image"

    echo "All resources have been cleaned up."
else
    echo "Resources were not cleaned up. You can manually clean up later with:"
    echo "aws ecr batch-delete-image --repository-name hello-repository --image-
ids imageTag=latest"
    echo "aws ecr delete-repository --repository-name hello-repository --force"
    echo "docker rmi hello-world:latest"
    echo "docker rmi $REPO_URI:latest"
fi

echo "================================================="
echo "Tutorial completed!"
echo "Log file: $LOG_FILE"
echo "================================================="
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - BatchDeleteImage
  - CreateRepository
  - DeleteRepository
  - GetCallerIdentity
  - GetLoginPassword

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan pesan internet of things

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat sumber daya IoT

- Konfigurasikan perangkat Anda
- Jalankan aplikasi sampel
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS IoT Core Getting Started Script
# This script creates AWS IoT resources, configures a device, and runs a sample
 application

# Set up logging
LOG_FILE="iot-core-setup.log"
echo "Starting AWS IoT Core setup at $(date)" > $LOG_FILE

# Function to log commands and their outputs
log_cmd() {
    echo "$(date): Running command: $1" >> $LOG_FILE
    eval "$1" 2>&1 | tee -a $LOG_FILE
    return ${PIPESTATUS[0]}
}

# Function to check for errors
check_error() {
    if [ $1 -ne 0 ]; then
        echo "ERROR: Command failed with exit code $1" | tee -a $LOG_FILE
        echo "Please check the log file $LOG_FILE for details" | tee -a $LOG_FILE
        cleanup_on_error
        exit $1
    fi
}
```

```
# Function to cleanup resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..." | tee -a
 $LOG_FILE
    echo "Resources created:" | tee -a $LOG_FILE
    if [ ! -z "$CERTIFICATE_ARN" ]; then
        echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
        if [ ! -z "$POLICY_NAME" ]; then
            log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
        fi
        if [ ! -z "$THING_NAME" ]; then
            log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --
principal $CERTIFICATE_ARN"
        fi
        if [ ! -z "$CERTIFICATE_ID" ]; then
            log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID
 --new-status INACTIVE"
            log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"
        fi
    fi
    if [ ! -z "$THING_NAME" ]; then
        echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-thing --thing-name $THING_NAME"
    fi
    if [ ! -z "$POLICY_NAME" ]; then
        echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    fi
    if [ ! -z "$SHARED_POLICY_NAME" ]; then
        echo "Shared Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"
    fi
}

# Generate unique identifiers
RANDOM_SUFFIX=$(openssl rand -hex 4)
THING_NAME="MyIoTThing-${RANDOM_SUFFIX}"
POLICY_NAME="MyIoTPolicy-${RANDOM_SUFFIX}"
SHARED_POLICY_NAME="SharedSubPolicy-${RANDOM_SUFFIX}"
CERTS_DIR="$HOME/certs"

echo "================================================" | tee -a $LOG_FILE
echo "AWS IoT Core Getting Started" | tee -a $LOG_FILE
```

```
echo "================================================" | tee -a $LOG_FILE
echo "This script will:" | tee -a $LOG_FILE
echo "1. Create AWS IoT resources (policy, thing, certificate)" | tee -a
 $LOG_FILE
echo "2. Configure your device" | tee -a $LOG_FILE
echo "3. Set up for running the sample application" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Certificates Directory: $CERTS_DIR" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE

# Get AWS account ID
echo "Getting AWS account ID..." | tee -a $LOG_FILE
ACCOUNT_ID=$(log_cmd "aws sts get-caller-identity --query Account --output text")
check_error $?

# Get AWS region
echo "Getting AWS region..." | tee -a $LOG_FILE
REGION=$(log_cmd "aws configure get region")
check_error $?
if [ -z "$REGION" ]; then
    echo "AWS region not configured. Please run 'aws configure' to set your
 region." | tee -a $LOG_FILE
    exit 1
fi

echo "Using AWS Account ID: $ACCOUNT_ID and Region: $REGION" | tee -a $LOG_FILE

# Step 1: Create AWS IoT Resources
echo "" | tee -a $LOG_FILE
echo "Step 1: Creating AWS IoT Resources..." | tee -a $LOG_FILE

# Create IoT policy
echo "Creating IoT policy document..." | tee -a $LOG_FILE
cat > iot-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
```

```
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/test-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic"
      ]
    }
  ]
}
EOF

echo "Creating IoT policy: $POLICY_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-policy --policy-name $POLICY_NAME --policy-
document file://iot-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

# Create IoT thing
echo "Creating IoT thing: $THING_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-thing --thing-name $THING_NAME"
check_error $?

THING_ARN=$(aws iot describe-thing --thing-name "$THING_NAME" --query 'thingArn'
 --output text)
aws iot tag-resource --resource-arn "$THING_ARN" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-iot-core-gs
```

```
# Create directory for certificates
echo "Creating certificates directory..." | tee -a $LOG_FILE
log_cmd "mkdir -p $CERTS_DIR"
check_error $?

# Create keys and certificate
echo "Creating keys and certificate..." | tee -a $LOG_FILE
CERT_OUTPUT=$(log_cmd "aws iot create-keys-and-certificate --set-as-active
 --certificate-pem-outfile $CERTS_DIR/device.pem.crt --public-key-outfile
 $CERTS_DIR/public.pem.key --private-key-outfile $CERTS_DIR/private.pem.key")
check_error $?

# Extract certificate ARN and ID
CERTIFICATE_ARN=$(echo "$CERT_OUTPUT" | grep "certificateArn" | cut -d'"' -f4)
CERTIFICATE_ID=$(echo "$CERTIFICATE_ARN" | cut -d/ -f2)

if [ -z "$CERTIFICATE_ARN" ] || [ -z "$CERTIFICATE_ID" ]; then
    echo "Failed to extract certificate ARN or ID" | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE

# Attach policy to certificate
echo "Attaching policy to certificate..." | tee -a $LOG_FILE
log_cmd "aws iot attach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Attach certificate to thing
echo "Attaching certificate to thing..." | tee -a $LOG_FILE
log_cmd "aws iot attach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"
check_error $?

# Download Amazon Root CA certificate
echo "Downloading Amazon Root CA certificate..." | tee -a $LOG_FILE
log_cmd "curl -s -o $CERTS_DIR/Amazon-root-CA-1.pem https://www.amazontrust.com/
repository/AmazonRootCA1.pem"
check_error $?

# Step 2: Configure Your Device
```

```
echo "" | tee -a $LOG_FILE
echo "Step 2: Configuring Your Device..." | tee -a $LOG_FILE

# Check if Git is installed
echo "Checking if Git is installed..." | tee -a $LOG_FILE
if ! command -v git &> /dev/null; then
    echo "Git is not installed. Please install Git and run this script again." |
 tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Check if Python is installed
echo "Checking if Python is installed..." | tee -a $LOG_FILE
if ! command -v python3 &> /dev/null; then
    echo "Python 3 is not installed. Please install Python 3 and run this script
 again." | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Install AWS IoT Device SDK for Python
echo "Installing AWS IoT Device SDK for Python..." | tee -a $LOG_FILE
log_cmd "python3 -m pip install awsiotsdk"
check_error $?

# Clone the AWS IoT Device SDK for Python repository
echo "Cloning AWS IoT Device SDK for Python repository..." | tee -a $LOG_FILE
if [ ! -d "$HOME/aws-iot-device-sdk-python-v2" ]; then
    log_cmd "cd $HOME && git clone https://github.com/aws/aws-iot-device-sdk-
python-v2.git"
    check_error $?
else
    echo "AWS IoT Device SDK for Python repository already exists." | tee -a
 $LOG_FILE
fi

# Step 3: Get AWS IoT Endpoint
echo "" | tee -a $LOG_FILE
echo "Step 3: Getting AWS IoT Endpoint..." | tee -a $LOG_FILE

IOT_ENDPOINT=$(log_cmd "aws iot describe-endpoint --endpoint-type iot:Data-ATS --
query endpointAddress --output text")
check_error $?
```

```
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE

# Create a shared subscription policy (optional)
echo "" | tee -a $LOG_FILE
echo "Creating shared subscription policy (optional)..." | tee -a $LOG_FILE

cat > shared-sub-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic",
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/\$share/*/test/topic"
      ]
    }
  ]
}
EOF
```

```
log_cmd "aws iot create-policy --policy-name $SHARED_POLICY_NAME --policy-
document file://shared-sub-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

log_cmd "aws iot attach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Summary of created resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Setup Complete! Resources Created:" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Shared Subscription Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE
echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate Files Location: $CERTS_DIR" | tee -a $LOG_FILE
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE

# Instructions for running the sample application
echo "" | tee -a $LOG_FILE
echo "To run the sample application, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 pubsub.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "To run the shared subscription example, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 mqtt5_shared_subscription.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key \\" | tee -a $LOG_FILE
echo "  --group_identifier consumer" | tee -a $LOG_FILE

# Ask if user wants to clean up resources
echo "" | tee -a $LOG_FILE
```

```
echo "================================================" | tee -a $LOG_FILE
echo "CLEANUP CONFIRMATION" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Do you want to clean up all created resources? (y/n): " | tee -a $LOG_FILE
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a $LOG_FILE

    # Detach policies from certificate
    echo "Detaching policies from certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
    log_cmd "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"

    # Detach certificate from thing
    echo "Detaching certificate from thing..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"

    # Update certificate status to INACTIVE
    echo "Setting certificate to inactive..." | tee -a $LOG_FILE
    log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE"

    # Delete certificate
    echo "Deleting certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"

    # Delete thing
    echo "Deleting thing..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-thing --thing-name $THING_NAME"

    # Delete policies
    echo "Deleting policies..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"

    echo "Cleanup complete!" | tee -a $LOG_FILE
else
    echo "Resources were not cleaned up. You can manually clean them up later." |
 tee -a $LOG_FILE
    echo "To clean up resources, run the following commands:" | tee -a $LOG_FILE
```

```
    echo "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE" | tee -a $LOG_FILE
    echo "aws iot delete-certificate --certificate-id $CERTIFICATE_ID" | tee -a
 $LOG_FILE
    echo "aws iot delete-thing --thing-name $THING_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $POLICY_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $SHARED_POLICY_NAME" | tee -a
 $LOG_FILE
fi

echo "" | tee -a $LOG_FILE
echo "Script execution completed. See $LOG_FILE for details." | tee -a $LOG_FILE
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachPolicy
  - AttachThingPrincipal
  - CreateKeysAndCertificate
  - CreatePolicy
  - CreateThing
  - DeleteCertificate
  - DeletePolicy
  - DeleteThing
  - DescribeEndpoint
  - DetachPolicy
  - DetachThingPrincipal
  - GetCallerIdentity
  - UpdateCertificate

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan toko fitur pembelajaran mesin

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Siapkan izin IAM
- Buat peran SageMaker eksekusi
- Buat grup fitur
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
```

```
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
```

```
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
```

```
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
```

```
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi
```

```
# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
```

```
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
```

```
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
```

```
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
```

```
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"
```

```
ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
```

```
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
```

```
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "========================================="
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .

- AttachRolePolicy
- CreateBucket
- CreateFeatureGroup
- CreateRole
- DeleteBucket
- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan cluster kubernetes terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat VPC untuk kluster EKS Anda
- Buat peran IAM untuk kluster EKS Anda
- Buat kluster EKS Anda
- Konfigurasikan kubectl untuk berkomunikasi dengan klaster Anda
- Membuat grup simpul terkelola
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
```

```
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
```

```
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
```

```
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."
```

```
# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
```

```
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
```

```
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"
```

```
# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
```

```
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
```

```
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
```

```
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNodegroups
  - UpdateKubeconfig
  - Tunggu (CloudFormation)
  - Tunggu (Amazon EKS)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan streaming terkelola

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat kluster MSK
- Buat izin IAM untuk akses MSK

- Buat mesin klien
- Dapatkan broker bootstrap
- Siapkan mesin klien
- Buat topik dan produce/consume data
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
```

```
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}
```

```
# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"
```

```
                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
```

```
   if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
       echo "Terminating EC2 instance: $INSTANCE_ID"
       aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
to terminate instance"
       echo "Waiting for instance to terminate..."
       aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
"Failed to wait for instance termination"
   fi

   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
       aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
```

```
    fi

    # Remove role from instance profile
    if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
        echo "Removing role from instance profile"
        aws iam remove-role-from-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" \
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
delete policy"
    fi
```

```
    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done
```

```
    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
```

```
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
```

```
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
```

```
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
```

```
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
```

```
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"
```

```
# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"
```

```
# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
```

```
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
```

```
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)
```

```
if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
```

```
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done
```

```
echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
```

```
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
```

```
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="
```

```
# Display summary of created resources
echo ""
echo "================================================"
echo "RESOURCE SUMMARY"
echo "================================================"
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "================================================"
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
```

```
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
```

```
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster
  - DeleteInstanceProfile

- DeleteKeyPair
- DeletePolicy
- DeleteRole
- DeleteSecurityGroup
- DescribeAvailabilityZones
- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan pemberitahuan push

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Membuat aplikasi
- Aktifkan saluran pemberitahuan push
- Kirim pemberitahuan push
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS End User Messaging Push Getting Started Script
# This script creates an AWS End User Messaging Push application and demonstrates
# how to enable push notification channels and send a test message.
#
# Prerequisites:
# - AWS CLI installed and configured
# - Appropriate IAM permissions for Pinpoint operations
#
# Usage: ./aws-end-user-messaging-gs.sh [--auto-cleanup]

set -uo pipefail

# Security: Set secure umask for created files
umask 0077

# Set up logging
LOG_DIR="./aws-eump-logs"
mkdir -p "$LOG_DIR"
LOG_FILE="$LOG_DIR/aws-end-user-messaging-push-script-$(date +%Y%m%d-%H%M%S).log"
```

```
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS End User Messaging Push setup script..."
echo "Logging to $LOG_FILE"
echo "Timestamp: $(date)"

# Security: Track created resources for cleanup
declare -a TEMP_FILES=()
declare -a AWS_RESOURCES=()

# Cleanup function with improved security
cleanup() {
    local exit_code=$?
    echo "Cleaning up temporary resources..."

    # Remove temporary files securely
    for temp_file in "${TEMP_FILES[@]+"${TEMP_FILES[@]}"}"; do
        if [ -f "$temp_file" ]; then
            shred -vfz -n 3 "$temp_file" 2>/dev/null || rm -f "$temp_file"
        fi
    done

    # Optionally delete AWS resources
    if [ "${DELETE_AWS_RESOURCES:-false}" = "true" ]; then
        for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
            echo "Deleting AWS resource: $resource"
            aws pinpoint delete-app --application-id "$resource" 2>/dev/null || \
                echo "Warning: Failed to delete application $resource"
        done
    fi

    exit "$exit_code"
}

trap cleanup EXIT INT TERM

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
    local ignore_error=${3:-false}

    if echo "$output" | grep -qi "error\|exception\|fail"; then
        echo "ERROR: Command failed: $cmd" >&2
```

```
        echo "Error details: $output" >&2

        if [ "$ignore_error" = "true" ]; then
            echo "Ignoring error and continuing..." >&2
            return 1
        else
            return 2
        fi
    fi

    return 0
}

# Function to validate AWS CLI is configured
validate_aws_cli() {
    echo "Validating AWS CLI configuration..."

    # Check if AWS CLI is installed
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed. Please install it first." >&2
        echo "Visit: https://docs.aws.amazon.com/cli/latest/userguide/getting-
started-install.html" >&2
        return 1
    fi

    # Check AWS CLI version
    AWS_VERSION=$(aws --version 2>&1 | head -n1)
    echo "AWS CLI version: $AWS_VERSION"

    # Verify credentials are set (check for credential env vars or config file)
    if ! aws sts get-caller-identity &> /dev/null; then
        echo "ERROR: AWS CLI credentials are not configured or invalid." >&2
        echo "Please configure credentials via environment variables, credential
 file, or 'aws configure'" >&2
        return 1
    fi

    # Get current AWS identity and region
    CALLER_IDENTITY=$(aws sts get-caller-identity)
    CURRENT_REGION=$(aws configure get region 2>/dev/null || echo "us-east-1")
    echo "AWS CLI configured for:"
    echo "$CALLER_IDENTITY"
    echo "Current region: $CURRENT_REGION"
    echo ""
```

```
    return 0
}

# Function to check if jq is available for JSON parsing
check_json_tools() {
    if command -v jq &> /dev/null; then
        USE_JQ=true
        echo "jq is available for JSON parsing"
    else
        USE_JQ=false
        echo "jq is not available, using grep for JSON parsing"
        echo "Consider installing jq for better JSON handling: https://
stedolan.github.io/jq/"
    fi
}

# Function to extract JSON values safely
extract_json_value() {
    local json=$1
    local key=$2

    if [ "$USE_JQ" = "true" ]; then
        echo "$json" | jq -r ".ApplicationResponse.$key // empty" 2>/dev/null ||
 echo ""
    else
        # Fallback to grep method with better validation
        echo "$json" | grep -o "\"$key\": \"[^\"]*" | cut -d'"' -f4 | head -n1 ||
 echo ""
    fi
}

# Function to validate required IAM permissions
validate_permissions() {
    echo "Validating IAM permissions..."

    # Test basic Pinpoint permissions
    if ! aws pinpoint get-apps > /dev/null 2>&1; then
        echo "WARNING: Unable to list Pinpoint applications." >&2
        echo "Please ensure you have appropriate IAM permissions for Pinpoint
 operations." >&2
        echo "Required permissions:" >&2
        echo "  - mobiletargeting:GetApps" >&2
        echo "  - mobiletargeting:CreateApp" >&2
```

```
        echo "  - mobiletargeting:DeleteApp" >&2
        echo "  - mobiletargeting:UpdateGcmChannel" >&2
        echo "  - mobiletargeting:UpdateApnsChannel" >&2
        echo "  - mobiletargeting:SendMessages" >&2
    else
        echo "Basic Pinpoint permissions validated."
    fi
}

# Function to validate input parameters
validate_input() {
    local app_name=$1

    # Validate app name length and characters
    if [ ${#app_name} -gt 64 ]; then
        echo "ERROR: Application name exceeds maximum length of 64 characters"
 >&2
        return 1
    fi

    if ! [[ "$app_name" =~ ^[a-zA-Z0-9_-]+$ ]]; then
        echo "ERROR: Application name contains invalid characters" >&2
        return 1
    fi

    return 0
}

# Function to create secure temporary files
create_temp_file() {
    local temp_file
    temp_file=$(mktemp) || {
        echo "ERROR: Failed to create temporary file" >&2
        return 1
    }
    chmod 600 "$temp_file"
    TEMP_FILES+=("$temp_file")
    echo "$temp_file"
}

# Validate prerequisites
if ! validate_aws_cli; then
    exit 1
fi
```

```
check_json_tools
validate_permissions

# Generate a random suffix for resource names to avoid conflicts
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n1)
APP_NAME="PushNotificationApp-${RANDOM_SUFFIX}"

# Validate input
if ! validate_input "$APP_NAME"; then
    exit 1
fi

echo "Creating application with name: $APP_NAME"

# Step 1: Create an application
echo "Executing: aws pinpoint create-app --create-application-request Name=
${APP_NAME}"
CREATE_APP_OUTPUT=$(aws pinpoint create-app --create-application-request "Name=
${APP_NAME},tags={project=doc-smith,tutorial=aws-end-user-messaging-gs}" 2>&1)

if ! check_error "$CREATE_APP_OUTPUT" "create-app"; then
    exit 1
fi

echo "Application created successfully:"
echo "$CREATE_APP_OUTPUT"

# Extract the application ID from the output
APP_ID=$(extract_json_value "$CREATE_APP_OUTPUT" "Id")

if [ -z "$APP_ID" ] || [ "$APP_ID" = "null" ]; then
    echo "ERROR: Failed to extract application ID from output" >&2
    echo "Output was: $CREATE_APP_OUTPUT" >&2
    exit 1
fi

echo "Application ID: $APP_ID"
AWS_RESOURCES+=("$APP_ID")

# Step 2: Enable FCM (GCM) channel with a sample API key
echo ""
echo "========================================="
echo "ENABLING FCM (GCM) CHANNEL"
```

```
echo "========================================="
echo "Note: This is using a placeholder API key for demonstration purposes only."
echo "In a production environment, you should use your actual FCM API key from
 Firebase Console."
echo ""
echo "IMPORTANT: The following command will likely fail because we're using a
 placeholder API key."
echo "This is expected behavior for this demonstration script."

echo "Executing: aws pinpoint update-gcm-channel --application-id $APP_ID --gcm-
channel-request ..."
UPDATE_GCM_OUTPUT=$(aws pinpoint update-gcm-channel \
    --application-id "$APP_ID" \
    --gcm-channel-request '{"Enabled": true, "ApiKey": "sample-fcm-api-key-for-
demo-only"}' 2>&1)

# We'll ignore this specific error since we're using a placeholder API key
if check_error "$UPDATE_GCM_OUTPUT" "update-gcm-channel" "true"; then
    echo "FCM channel enabled successfully:"
    echo "$UPDATE_GCM_OUTPUT"
else
    echo "As expected, FCM channel update failed with the placeholder API key."
    echo "Error details: $UPDATE_GCM_OUTPUT"
    echo ""
    echo "To enable FCM in production:"
    echo "1. Go to Firebase Console (https://console.firebase.google.com/)"
    echo "2. Create or select your project"
    echo "3. Go to Project Settings > Cloud Messaging"
    echo "4. Copy the Server Key"
    echo "5. Replace 'sample-fcm-api-key-for-demo-only' with your actual Server
 Key"
fi

# Step 3: Try to enable APNS channel (this will also fail without real
 certificates)
echo ""
echo "========================================="
echo "ENABLING APNS CHANNEL (OPTIONAL)"
echo "========================================="
echo "Attempting to enable APNS channel with placeholder certificate..."
echo "This will also fail without real APNS certificates, which is expected."

# Create a placeholder APNS configuration
```

```
echo "Executing: aws pinpoint update-apns-channel --application-id $APP_ID --
apns-channel-request ..."
UPDATE_APNS_OUTPUT=$(aws pinpoint update-apns-channel \
    --application-id "$APP_ID" \
    --apns-channel-request '{"Enabled": true, "Certificate": "placeholder-
certificate", "PrivateKey": "placeholder-private-key"}' 2>&1)

if check_error "$UPDATE_APNS_OUTPUT" "update-apns-channel" "true"; then
    echo "APNS channel enabled successfully:"
    echo "$UPDATE_APNS_OUTPUT"
else
    echo "As expected, APNS channel update failed with placeholder certificates."
    echo "Error details: $UPDATE_APNS_OUTPUT"
    echo ""
    echo "To enable APNS in production:"
    echo "1. Generate APNS certificates from Apple Developer Console"
    echo "2. Convert certificates to PEM format"
    echo "3. Use the actual certificate and private key in the update-apns-
channel command"
fi

# Step 4: Create message files for different platforms
echo ""
echo "=========================================="
echo "CREATING MESSAGE FILES"
echo "=========================================="

# Create FCM message file securely
GCM_MESSAGE_FILE=$(create_temp_file)
echo "Creating FCM message file..."
cat > "$GCM_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-FCM": {
      "ChannelType": "GCM"
    }
  },
  "MessageConfiguration": {
    "GCMMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an FCM
 notification.",
      "Priority": "normal",
      "SilentPush": false,
```

```
      "Title": "My First FCM Push Notification",
      "TimeToLive": 30,
      "Data": {
        "key1": "value1",
        "key2": "value2"
      }
    }
  }
}
EOF

# Create APNS message file securely
APNS_MESSAGE_FILE=$(create_temp_file)
echo "Creating APNS message file..."
cat > "$APNS_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-APNS": {
      "ChannelType": "APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an APNS
 notification.",
      "Priority": "normal",
      "SilentPush": false,
      "Title": "My First APNS Push Notification",
      "TimeToLive": 30,
      "Badge": 1,
      "Sound": "default"
    }
  }
}
EOF

echo "Message files created:"
echo "- FCM message file (for FCM/Android)"
echo "- APNS message file (for APNS/iOS)"
echo ""
echo "Note: These messages use placeholder device tokens and will not actually be
 delivered."
```

```
echo "To send real messages, you would need to replace the sample device tokens
 with actual ones."

# Step 5: Demonstrate how to send messages (this will fail with placeholder
 tokens)
echo ""
echo "========================================="
echo "DEMONSTRATING MESSAGE SENDING"
echo "========================================="
echo "Attempting to send FCM message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<gcm-message>"
SEND_FCM_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$GCM_MESSAGE_FILE" 2>&1)

if check_error "$SEND_FCM_OUTPUT" "send-messages (FCM)" "true"; then
    echo "FCM message sent successfully:"
    echo "$SEND_FCM_OUTPUT"
else
    echo "As expected, FCM message sending failed with placeholder token."
    echo "Error details: $SEND_FCM_OUTPUT"
fi

echo ""
echo "Attempting to send APNS message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<apns-message>"
SEND_APNS_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$APNS_MESSAGE_FILE" 2>&1)

if check_error "$SEND_APNS_OUTPUT" "send-messages (APNS)" "true"; then
    echo "APNS message sent successfully:"
    echo "$SEND_APNS_OUTPUT"
else
    echo "As expected, APNS message sending failed with placeholder token."
    echo "Error details: $SEND_APNS_OUTPUT"
fi

# Step 6: Show application details
echo ""
```

```
echo "============================================="
echo "APPLICATION DETAILS"
echo "============================================="
echo "Retrieving application details..."

echo "Executing: aws pinpoint get-app --application-id $APP_ID"
GET_APP_OUTPUT=$(aws pinpoint get-app --application-id "$APP_ID" 2>&1)
if check_error "$GET_APP_OUTPUT" "get-app"; then
    echo "Application details:"
    echo "$GET_APP_OUTPUT"
fi

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
echo "AWS Resources:"
for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
    echo "- Application: $resource"
done

echo ""
echo "Files created:"
echo "- $LOG_FILE (script log)"

# Auto-cleanup information
echo ""
echo "============================================="
echo "CLEANUP INFORMATION"
echo "============================================="
echo "This script created AWS resources that may incur charges."
echo "AWS resources will be automatically cleaned up on script exit."
echo ""
echo "To manually delete resources later, use:"
echo "  aws pinpoint delete-app --application-id $APP_ID"

# Set flag to delete AWS resources on cleanup
DELETE_AWS_RESOURCES=true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
```

```
echo "This script demonstrated:"
echo "1. Creating an AWS End User Messaging Push application"
echo "2. Attempting to enable FCM and APNS channels (with placeholder
 credentials)"
echo "3. Creating message templates for different platforms"
echo "4. Demonstrating message sending commands (with placeholder tokens)"
echo "5. Retrieving application details"
echo "6. Proper cleanup of resources"
echo ""
echo "Security best practices implemented:"
echo "- Secure temporary file handling with restricted permissions"
echo "- Input validation for application names"
echo "- Error handling with proper exit codes"
echo "- Credential validation before AWS operations"
echo "- Automatic cleanup on script exit"
echo "- Secure file destruction for sensitive data"
echo ""
echo "For production use:"
echo "- Replace placeholder API keys with real FCM server keys"
echo "- Replace placeholder certificates with real APNS certificates"
echo "- Replace placeholder device tokens with real device tokens"
echo "- Use AWS IAM roles instead of long-term credentials"
echo "- Implement comprehensive error handling"
echo "- Store sensitive credentials in AWS Secrets Manager or Parameter Store"
echo "- Enable CloudTrail for audit logging"
echo "- Use VPC endpoints for private AWS API access"
echo ""
echo "Log file: $LOG_FILE"
echo "Script completed at: $(date)"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - CreateApp
  - DeleteApp
  - GetApp
  - GetApps
  - GetCallerIdentity
  - SendMessages
  - UpdateApnsChannel
  - UpdateGcmChannel

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan mesin pencari dan analitik

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat domain OpenSearch Layanan
- Unggah data ke domain Anda
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Amazon OpenSearch Service Getting Started Script - Version 8 Fixed
# This script creates an OpenSearch domain, uploads data, searches documents, and
 cleans up resources
# Based on the tested and working 4-tutorial-final.md

# SECURITY IMPROVEMENTS IN THIS VERSION:
# 1. Removed hardcoded passwords - now generated securely
# 2. Improved access policy to use principle of least privilege
# 3. Added credential masking in logs
# 4. Removed credentials from command output
# 5. Added input validation for AWS region
# 6. Improved error handling and validation
# 7. Added secure temporary file handling
# 8. Removed unnecessary curl insecurity flags
# 9. Added better secret management practices
# 10. Improved resource tagging for better governance
```

```
set -e  # Exit on any error

# Set up logging with restricted permissions
LOG_FILE="opensearch_tutorial_v8_fixed.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon OpenSearch Service tutorial script v8-fixed at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Track if domain was successfully created
DOMAIN_CREATED=false
DOMAIN_ACTIVE=false

# Secure temporary directory
TEMP_DIR=$(mktemp -d)
trap 'rm -rf "$TEMP_DIR"' EXIT
chmod 700 "$TEMP_DIR"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    if [[ "$DOMAIN_CREATED" == "true" ]]; then
        echo "Checking if domain $DOMAIN_NAME exists before attempting to
 delete..."

        # Check if domain exists before trying to delete
        if aws opensearch describe-domain --domain-name "$DOMAIN_NAME" > /dev/
null 2>&1; then
            echo "Domain $DOMAIN_NAME exists. Proceeding with deletion."
            aws opensearch delete-domain --domain-name "$DOMAIN_NAME"
            echo "Domain deletion initiated. This may take several minutes to
 complete."
```

```
        else
            echo "Domain $DOMAIN_NAME does not exist or is not accessible. No
 deletion needed."
        fi
    else
        echo "No domain was successfully created. Nothing to clean up."
    fi

    # Securely clean up sensitive files
    if [[ -d "$TEMP_DIR" ]]; then
        shred -vfz -n 3 "$TEMP_DIR"/* 2>/dev/null || true
        rm -rf "$TEMP_DIR"
    fi
}

# Generate a random identifier for resource names to avoid conflicts
RANDOM_ID=$(openssl rand -hex 4)
DOMAIN_NAME="movies-${RANDOM_ID}"
MASTER_USER="master-user"

# Generate secure password using openssl instead of hardcoding
MASTER_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-25)

# Mask password in logs
echo "Using domain name: $DOMAIN_NAME"
echo "Using master username: $MASTER_USER"
echo "Using master password: [REDACTED]"

# Get AWS account ID (matches tutorial)
echo "Retrieving AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ $? -ne 0 ]] || [[ -z "$ACCOUNT_ID" ]]; then
    handle_error "Failed to retrieve AWS account ID. Please check your AWS
 credentials."
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Get current region (matches tutorial)
echo "Retrieving current AWS region..."
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION"
else
```

```
    echo "Using AWS region: $AWS_REGION"
fi

# Validate AWS region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    handle_error "Invalid AWS region format: $AWS_REGION"
fi

# Step 1: Create an OpenSearch Service Domain
echo "Creating OpenSearch Service domain..."
echo "This may take 15-30 minutes to complete."

# SECURITY IMPROVED: Use least privilege access policy
# This policy restricts access to specific actions and should be further
 restricted to specific principals in production
# Create the domain using CloudFormation (handles the 15-20 min creation wait)
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
STACK_NAME="tutorial-opensearch-${RANDOM_ID}"

echo "Creating domain $DOMAIN_NAME via CloudFormation..."
echo "This typically takes 15-20 minutes. CloudFormation will wait for
 completion."
aws cloudformation deploy \
  --template-file "$SCRIPT_DIR/cfn-opensearch-domain.yaml" \
  --stack-name "$STACK_NAME" \
  --parameter-overrides \
    DomainName="$DOMAIN_NAME" \
    MasterUserName="$MASTER_USER" \
    MasterUserPassword="$MASTER_PASSWORD" \
  --tags project=doc-smith tutorial=opensearch-service-gs \
  --no-fail-on-empty-changeset 2>&1 || handle_error "CloudFormation stack
 creation failed"

DOMAIN_CREATED=true
echo "Domain created successfully via CloudFormation."

# Get domain endpoint from stack outputs
DOMAIN_ENDPOINT=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query 'Stacks[0].Outputs[?OutputKey==`DomainEndpoint`].OutputValue' \
  --output text)

if [[ -z "$DOMAIN_ENDPOINT" ]] || [[ "$DOMAIN_ENDPOINT" == "None" ]]; then
    handle_error "Failed to get domain endpoint from CloudFormation outputs"
```

```
fi

echo "Domain endpoint: $DOMAIN_ENDPOINT"

# Wait for fine-grained access control to be fully ready
echo "Waiting for fine-grained access control to initialize..."
sleep 120

# Verify variables are set correctly (matches tutorial)
echo "Verifying configuration..."
echo "Domain endpoint: $DOMAIN_ENDPOINT"
echo "Master user: $MASTER_USER"
echo "Password set: $(if [ -n "$MASTER_PASSWORD" ]; then echo "Yes"; else echo
 "No"; fi)"

# Step 2: Upload Data to the Domain
echo "Preparing to upload data to the domain..."

# Create a file for the single document (matches tutorial exactly)
echo "Creating single document JSON file..."
SINGLE_MOVIE_FILE="$TEMP_DIR/single_movie.json"
cat > "$SINGLE_MOVIE_FILE" << 'EOF'
{
  "director": "Burton, Tim",
  "genre": ["Comedy","Sci-Fi"],
  "year": 1996,
  "actor": ["Jack Nicholson","Pierce Brosnan","Sarah Jessica Parker"],
  "title": "Mars Attacks!"
}
EOF
chmod 600 "$SINGLE_MOVIE_FILE"

# Create a file for bulk documents (matches tutorial exactly)
echo "Creating bulk documents JSON file..."
BULK_MOVIES_FILE="$TEMP_DIR/bulk_movies.json"
cat > "$BULK_MOVIES_FILE" << 'EOF'
{ "index" : { "_index": "movies", "_id" : "2" } }
{"director": "Frankenheimer, John", "genre": ["Drama", "Mystery", "Thriller",
 "Crime"], "year": 1962, "actor": ["Lansbury, Angela", "Sinatra, Frank", "Leigh,
 Janet", "Harvey, Laurence", "Silva, Henry", "Frees, Paul", "Gregory, James",
 "Bissell, Whit", "McGiver, John", "Parrish, Leslie", "Edwards, James", "Flowers,
 Bess", "Dhiegh, Khigh", "Payne, Julie", "Kleeb, Helen", "Gray, Joe", "Nalder,
 Reggie", "Stevens, Bert", "Masters, Michael", "Lowell, Tom"], "title": "The
 Manchurian Candidate"}
```

```
{ "index" : { "_index": "movies", "_id" : "3" } }
{"director": "Baird, Stuart", "genre": ["Action", "Crime", "Thriller"], "year":
 1998, "actor": ["Downey Jr., Robert", "Jones, Tommy Lee", "Snipes, Wesley",
 "Pantoliano, Joe", "Jacob, Irène", "Nelligan, Kate", "Roebuck, Daniel",
 "Malahide, Patrick", "Richardson, LaTanya", "Wood, Tom", "Kosik, Thomas",
 "Stellate, Nick", "Minkoff, Robert", "Brown, Spitfire", "Foster, Reese",
 "Spielbauer, Bruce", "Mukherji, Kevin", "Cray, Ed", "Fordham, David", "Jett,
 Charlie"], "title": "U.S. Marshals"}
{ "index" : { "_index": "movies", "_id" : "4" } }
{"director": "Ray, Nicholas", "genre": ["Drama", "Romance"], "year": 1955,
 "actor": ["Hopper, Dennis", "Wood, Natalie", "Dean, James", "Mineo, Sal",
 "Backus, Jim", "Platt, Edward", "Ray, Nicholas", "Hopper, William", "Allen,
 Corey", "Birch, Paul", "Hudson, Rochelle", "Doran, Ann", "Hicks, Chuck", "Leigh,
 Nelson", "Williams, Robert", "Wessel, Dick", "Bryar, Paul", "Sessions, Almira",
 "McMahon, David", "Peters Jr., House"], "title": "Rebel Without a Cause"}
EOF
chmod 600 "$BULK_MOVIES_FILE"

# Check if curl is installed
if ! command -v curl &> /dev/null; then
    echo "Warning: curl is not installed. Skipping data upload and search steps."
    echo "You can manually upload the data later using the commands in the
 tutorial."
else
    # IMPROVED: Test authentication with multiple approaches
    echo "Testing authentication with the OpenSearch domain..."
    echo "This test checks if fine-grained access control is ready for data
 operations."

    # Create credentials file for secure handling
    CREDENTIALS_FILE="$TEMP_DIR/.credentials"
    echo "$MASTER_USER:$MASTER_PASSWORD" > "$CREDENTIALS_FILE"
    chmod 600 "$CREDENTIALS_FILE"

    # Test 1: Basic authentication with root endpoint
    echo "Testing basic authentication with root endpoint..."
    AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
        --netrc-file "$CREDENTIALS_FILE" \
        --request GET \
        "https://${DOMAIN_ENDPOINT}/" 2>&1)

    echo "Basic auth test result:"
    echo "$AUTH_TEST_RESULT"
```

```
    # Extract HTTP status code
    HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)

    # Function to check if HTTP code is 2xx
    is_success_code() {
        local code=$1
        if [[ "$code" =~ ^2[0-9][0-9]$ ]]; then
            return 0
        else
            return 1
        fi
    }

    # Check if basic authentication test was successful (200 or 2xx status codes)
    if is_success_code "$HTTP_CODE"; then
        echo "# Basic authentication test successful! (HTTP $HTTP_CODE)"
        AUTH_SUCCESS=true
        AUTH_METHOD="basic"
    else
        echo "Basic authentication failed with HTTP code: $HTTP_CODE"

        # Test 2: Try cluster health endpoint
        echo "Testing with cluster health endpoint..."
        HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

        echo "Cluster health test result:"
        echo "$HEALTH_TEST_RESULT"

        HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

        if is_success_code "$HEALTH_HTTP_CODE"; then
            echo "# Cluster health authentication test successful! (HTTP
 $HEALTH_HTTP_CODE)"
            AUTH_SUCCESS=true
            AUTH_METHOD="basic"
        else
            echo "Cluster health authentication also failed with HTTP code:
 $HEALTH_HTTP_CODE"

            # Check for specific error patterns
```

```
            if echo "$AUTH_TEST_RESULT" | grep -q "anonymous is not authorized";
 then
                echo "Error: Request is being treated as anonymous
 (authentication not working)"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Unauthorized"; then
                echo "Error: Authentication credentials rejected"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Forbidden"; then
                echo "Error: Authentication succeeded but access is forbidden"
            fi

            echo "Waiting additional time and retrying with exponential
 backoff..."

            # Retry authentication test with exponential backoff
            AUTH_RETRY_COUNT=0
            MAX_AUTH_RETRIES=5
            WAIT_TIME=60
            AUTH_SUCCESS=false

            while [[ $AUTH_RETRY_COUNT -lt $MAX_AUTH_RETRIES ]]; do
                echo "Retrying authentication test (attempt $((AUTH_RETRY_COUNT
+1))/$MAX_AUTH_RETRIES) after ${WAIT_TIME} seconds..."
                sleep $WAIT_TIME

                # Try both endpoints
                AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/" 2>&1)

                HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

                echo "Retry result (HTTP $HTTP_CODE):"
                echo "$AUTH_TEST_RESULT"

                if is_success_code "$HTTP_CODE"; then
                    echo "# Authentication test successful after retry! (HTTP
 $HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi
```

```
                # Also try cluster health
                HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

                HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:"
| cut -d: -f2)

                if is_success_code "$HEALTH_HTTP_CODE"; then
                    echo "# Cluster health authentication successful after retry!
(HTTP $HEALTH_HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi

                AUTH_RETRY_COUNT=$((AUTH_RETRY_COUNT+1))
                # Exponential backoff: double the wait time each retry (max 10
minutes)
                WAIT_TIME=$((WAIT_TIME * 2))
                if [[ $WAIT_TIME -gt 600 ]]; then
                    WAIT_TIME=600
                fi
            done
        fi
    fi

    # Proceed with data operations if authentication is working
    if [[ "$AUTH_SUCCESS" == "true" ]]; then
        echo "Authentication successful using $AUTH_METHOD method. Proceeding
with data operations."

        # Upload single document (matches tutorial exactly)
        echo "Uploading single document..."
        UPLOAD_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request PUT \
            --header 'Content-Type: application/json' \
            --data @"$SINGLE_MOVIE_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_doc/1" 2>&1)

        echo "Upload response:"
        echo "$UPLOAD_RESULT"
```

```
        UPLOAD_HTTP_CODE=$(echo "$UPLOAD_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        if is_success_code "$UPLOAD_HTTP_CODE" && echo "$UPLOAD_RESULT" | grep -q
 '"result"'; then
            echo "# Single document uploaded successfully! (HTTP
 $UPLOAD_HTTP_CODE)"
        else
            echo "# Warning: Single document upload may have failed (HTTP
 $UPLOAD_HTTP_CODE)"
        fi

        # Upload bulk documents (matches tutorial exactly)
        echo "Uploading bulk documents..."
        BULK_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request POST \
            --header 'Content-Type: application/x-ndjson' \
            --data-binary @"$BULK_MOVIES_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_bulk" 2>&1)

        echo "Bulk upload response:"
        echo "$BULK_RESULT"

        BULK_HTTP_CODE=$(echo "$BULK_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)
        if is_success_code "$BULK_HTTP_CODE" && echo "$BULK_RESULT" | grep -q
 '"errors": false'; then
            echo "# Bulk documents uploaded successfully! (HTTP $BULK_HTTP_CODE)"
        else
            echo "# Warning: Bulk document upload may have failed (HTTP
 $BULK_HTTP_CODE)"
        fi

        # Wait a moment for indexing
        echo "Waiting for documents to be indexed..."
        sleep 5

        # Step 3: Search Documents (matches tutorial exactly)
        echo "Searching for documents containing 'mars'..."
        SEARCH_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true" 2>&1)
```

```
        SEARCH_HTTP_CODE=$(echo "$SEARCH_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        echo "Search results for 'mars' (HTTP $SEARCH_HTTP_CODE):"
        echo "$SEARCH_RESULT"

        echo "Searching for documents containing 'rebel'..."
        REBEL_SEARCH=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=rebel&pretty=true" 2>&1)

        REBEL_HTTP_CODE=$(echo "$REBEL_SEARCH" | grep "HTTP_CODE:" | cut -d: -f2)
        echo "Search results for 'rebel' (HTTP $REBEL_HTTP_CODE):"
        echo "$REBEL_SEARCH"

        # Verify search results
        if is_success_code "$SEARCH_HTTP_CODE" && echo "$SEARCH_RESULT" | grep -q
 '"hits"'; then
            echo "# Search functionality is working!"
        else
            echo "# Warning: Search may not be working properly."
        fi

    else
        echo ""
        echo "========================================="
        echo "AUTHENTICATION TROUBLESHOOTING"
        echo "========================================="
        echo "Authentication failed after all retries. This may be due to:"
        echo "1. Fine-grained access control not fully initialized (most common)"
        echo "2. Domain configuration issues"
        echo "3. Network connectivity issues"
        echo "4. AWS credentials or permissions issues"
        echo ""
        echo "DOMAIN CONFIGURATION DEBUG:"
        echo "Let's check the domain configuration..."

        # Debug domain configuration
        DOMAIN_CONFIG=$(aws opensearch describe-domain --domain-
name "$DOMAIN_NAME" --query 'DomainStatus.{AdvancedSecurityOptions:
 AdvancedSecurityOptions, AccessPolicies: AccessPolicies}' --output json 2>&1)
        echo "Domain configuration:"
        echo "$DOMAIN_CONFIG"
```

```
        echo ""
        echo "MANUAL TESTING COMMANDS:"
        echo "You can try these commands manually in 10-15 minutes:"
        echo ""
        echo "# Test basic authentication:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/\""
        echo ""
        echo "# Test cluster health:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/_cluster/health\""
        echo ""
        echo "# Upload single document:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" --request PUT --
header 'Content-Type: application/json' --data @single_movie.json \"https://
${DOMAIN_ENDPOINT}/movies/_doc/1\""
        echo ""
        echo "# Search for documents:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true\""
        echo ""
        echo "TROUBLESHOOTING TIPS:"
        echo "- Wait 10-15 more minutes and try the manual commands"
        echo "- Check AWS CloudTrail logs for authentication errors"
        echo "- Verify your AWS region is correct: $AWS_REGION"
        echo "- Ensure your AWS credentials have OpenSearch permissions"
        echo "- Try accessing OpenSearch Dashboards to verify the master user
 works"
        echo ""
        echo "Skipping data upload and search operations for now."
        echo "The domain is created and accessible via OpenSearch Dashboards."
    fi

    # Securely remove credentials file
    shred -vfz -n 3 "$CREDENTIALS_FILE" 2>/dev/null || true
fi

# Display OpenSearch Dashboards URL (matches tutorial)
echo ""
echo "============================================"
echo "OPENSEARCH DASHBOARDS ACCESS"
echo "============================================"
echo "OpenSearch Dashboards URL: https://${DOMAIN_ENDPOINT}/_dashboards/"
echo "Username: $MASTER_USER"
```

```
echo "Password: [REDACTED - stored securely]"
echo ""
echo "You can access OpenSearch Dashboards using the credentials provided during
 domain creation."
echo "If you uploaded data successfully, you can create an index pattern for
 'movies'."
echo ""

# Summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
echo "OpenSearch Domain Name: $DOMAIN_NAME"
echo "OpenSearch Domain Endpoint: $DOMAIN_ENDPOINT"
echo "AWS Region: $AWS_REGION"
echo "Master Username: $MASTER_USER"
echo ""
echo "ESTIMATED COST: ~$0.038/hour (~$0.91/day) until deleted"
echo ""
echo "Make sure to save the domain name and endpoint for future reference."
echo ""

# Auto-confirm cleanup: Yes, clean up all resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Automatically cleaning up all created resources..."
CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."
    if [ -n "${STACK_NAME:-}" ]; then
        echo "Deleting CloudFormation stack $STACK_NAME..."
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "# Stack deletion initiated. This may take several minutes."
        echo "  Monitor: aws cloudformation describe-stacks --stack-name
 $STACK_NAME"
    else
        aws opensearch delete-domain --domain-name "$DOMAIN_NAME" 2>/dev/null
        echo "# Domain deletion initiated."
    fi
else
```

```
    echo "Resources will NOT be deleted automatically."
    echo ""
    echo "To delete later:"
    echo "  aws cloudformation delete-stack --stack-name ${STACK_NAME:-tutorial-
opensearch-XXXX}"
    echo ""
    echo "# IMPORTANT: Keeping these resources will incur ongoing AWS charges!"
fi

# Clean up temporary files (handled by trap)
echo "Cleaning up temporary files..."

# Disable the trap since we're handling cleanup manually
trap - EXIT

# Final cleanup
rm -rf "$TEMP_DIR" 2>/dev/null || true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
echo "Script completed at $(date)"
echo "All output has been logged to: $LOG_FILE"
echo ""
echo "Next steps:"
echo "1. Access OpenSearch Dashboards at: https://${DOMAIN_ENDPOINT}/
_dashboards/"
echo "2. Create visualizations and dashboards"
echo "3. Explore the OpenSearch API"
echo "4. Remember to delete resources when done to avoid charges"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - CreateDomain
  - DeleteDomain
  - DescribeDomain
  - GetCallerIdentity

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai dengan firewall aplikasi web

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat ACL web
- Tambahkan aturan pencocokan string
- Tambahkan aturan terkelola
- Mengonfigurasikan pencatatan log
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS WAF Getting Started Script
# This script creates a Web ACL with a string match rule and AWS Managed Rules,
# associates it with a CloudFront distribution, and then cleans up all resources.

set -euo pipefail

# Security: Restrict file permissions
umask 077

# Set up logging with secure file handling
LOG_FILE="waf-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS WAF Getting Started Tutorial"
echo "================================================="
```

```
echo "This script will create AWS WAF resources and associate"
echo "them with a CloudFront distribution."
echo ""

# Maximum number of retries for operations
MAX_RETRIES=3

# Function to handle errors securely
handle_error() {
    local error_msg="$1"
    echo "ERROR: $error_msg" >&2
    echo "Check the log file for details: $LOG_FILE" >&2
    cleanup_resources
    exit 1
}

# Function to validate AWS CLI JSON output
validate_json() {
    local json_string="$1"
    if ! echo "$json_string" | jq empty 2>/dev/null; then
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
extract_json_value() {
    local json_string="$1"
    local key_path="$2"

    if ! validate_json "$json_string"; then
        return 1
    fi

    echo "$json_string" | jq -r "$key_path" 2>/dev/null || return 1
}

# Function to clean up resources securely
cleanup_resources() {
    echo ""
    echo "=================================================="
    echo "CLEANING UP RESOURCES"
    echo "=================================================="
```

```
    if [ -n "${DISTRIBUTION_ID:-}" ] && [ -n "${WEB_ACL_ARN:-}" ]; then
        echo "Disassociating Web ACL from CloudFront distribution..."
        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID"
            return
        fi

        local disassociate_result
        disassociate_result=$(aws wafv2 disassociate-web-acl \
            --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
            --region us-east-1 2>&1) || true

        if echo "$disassociate_result" | grep -qi "error"; then
            echo "Warning: Failed to disassociate Web ACL: $disassociate_result"
        else
            echo "Web ACL disassociated successfully."
        fi
    fi

    if [ -n "${WEB_ACL_ID:-}" ] && [ -n "${WEB_ACL_NAME:-}" ]; then
        echo "Deleting Web ACL..."

        local get_result
        get_result=$(aws wafv2 get-web-acl \
            --name "$WEB_ACL_NAME" \
            --scope CLOUDFRONT \
            --id "$WEB_ACL_ID" \
            --region us-east-1 2>&1) || true

        if echo "$get_result" | grep -qi "error"; then
            echo "Warning: Failed to get Web ACL for deletion: $get_result"
            echo "You may need to manually delete the Web ACL using the AWS
 Console."
        else
            local latest_token
            latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken'
 2>/dev/null) || latest_token=""

            if [ -n "$latest_token" ]; then
```

```
                local delete_result
                delete_result=$(aws wafv2 delete-web-acl \
                    --name "$WEB_ACL_NAME" \
                    --scope CLOUDFRONT \
                    --id "$WEB_ACL_ID" \
                    --lock-token "$latest_token" \
                    --region us-east-1 2>&1) || true

                if echo "$delete_result" | grep -qi "error"; then
                    echo "Warning: Failed to delete Web ACL: $delete_result"
                    echo "You may need to manually delete the Web ACL using the
 AWS Console."
                else
                    echo "Web ACL deleted successfully."
                fi
            else
                echo "Warning: Could not extract lock token for deletion. You may
 need to manually delete the Web ACL."
            fi
        fi
    fi

    echo "Cleanup process completed."
}

# Security: Trap EXIT to ensure cleanup on any exit
trap cleanup_resources EXIT

# Generate a random identifier for resource names using secure method
RANDOM_ID=$(openssl rand -hex 4) || handle_error "Failed to generate random ID"
WEB_ACL_NAME="MyWebACL-${RANDOM_ID}"
METRIC_NAME="MyWebACLMetrics-${RANDOM_ID}"

echo "Using Web ACL name: $WEB_ACL_NAME"

# Step 1: Create a Web ACL
echo ""
echo "=================================================="
echo "STEP 1: Creating Web ACL"
echo "=================================================="

local create_result
create_result=$(aws wafv2 create-web-acl \
    --name "$WEB_ACL_NAME" \
```

```
    --scope "CLOUDFRONT" \
    --default-action Allow={} \
    --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-waf-gs \
    --region us-east-1 2>&1) || handle_error "Failed to create Web ACL"

if ! validate_json "$create_result"; then
    handle_error "Invalid JSON response from create-web-acl"
fi

# Extract Web ACL ID, ARN, and Lock Token from the response
WEB_ACL_ID=$(extract_json_value "$create_result" '.Summary.Id') || handle_error
 "Failed to extract Web ACL ID"
WEB_ACL_ARN=$(extract_json_value "$create_result" '.Summary.ARN') || handle_error
 "Failed to extract Web ACL ARN"
LOCK_TOKEN=$(extract_json_value "$create_result" '.Summary.LockToken') ||
 handle_error "Failed to extract Lock Token"

echo "Web ACL created successfully with ID: $WEB_ACL_ID"
echo "Lock Token: [REDACTED]"

# Step 2: Add a String Match Rule
echo ""
echo "=================================================="
echo "STEP 2: Adding String Match Rule"
echo "=================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add string match rule..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
```

```
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
```

```
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add string match rule after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add string match rule: $update_result"
    else
        echo "String match rule added successfully."
        break
    fi
done

# Step 3: Add AWS Managed Rules
echo ""
```

```
echo "=================================================="
echo "STEP 3: Adding AWS Managed Rules"
echo "=================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add AWS Managed Rules..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
```

```
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        },
        {
            "Name": "AWS-AWSManagedRulesCommonRuleSet",
            "Priority": 1,
            "Statement": {
                "ManagedRuleGroupStatement": {
```

```
                    "VendorName": "AWS",
                    "Name": "AWSManagedRulesCommonRuleSet",
                    "ExcludedRules": []
                }
            },
            "OverrideAction": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add AWS Managed Rules after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add AWS Managed Rules: $update_result"
    else
        echo "AWS Managed Rules added successfully."
        break
    fi
done

# Step 4: List CloudFront distributions
echo ""
echo "=================================================="
echo "STEP 4: Listing CloudFront Distributions"
echo "=================================================="

local cf_result
```

```
cf_result=$(aws cloudfront list-distributions --query "DistributionList.Items[*].
{Id:Id,DomainName:DomainName}" --output table 2>&1) || cf_result=""

if echo "$cf_result" | grep -qi "error"; then
    echo "Warning: Failed to list CloudFront distributions: $cf_result"
    echo "Continuing without CloudFront association."
    DISTRIBUTION_ID=""
else
    echo "$cf_result"

    # Auto-select first CloudFront distribution if available
    echo ""
    echo "=================================================="
    echo "STEP 5: Associate Web ACL with CloudFront Distribution"
    echo "=================================================="

    local first_dist
    first_dist=$(aws cloudfront list-distributions --query
 "DistributionList.Items[0].Id" --output text 2>&1) || first_dist=""

    if [ -n "$first_dist" ] && [ "$first_dist" != "None" ] && ! echo
 "$first_dist" | grep -qi "error"; then
        DISTRIBUTION_ID="$first_dist"
        echo "Using CloudFront distribution: $DISTRIBUTION_ID"

        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID for association"
            DISTRIBUTION_ID=""
        else
            local associate_result
            associate_result=$(aws wafv2 associate-web-acl \
                --web-acl-arn "$WEB_ACL_ARN" \
                --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
                --region us-east-1 2>&1) || true

            if echo "$associate_result" | grep -qi "error"; then
                echo "Warning: Failed to associate Web ACL with CloudFront
 distribution: $associate_result"
                echo "Continuing without CloudFront association."
```

```
                DISTRIBUTION_ID=""
            else
                echo "Web ACL associated with CloudFront distribution
 successfully."
            fi
        fi
    else
        echo "No CloudFront distributions available. Skipping association."
        DISTRIBUTION_ID=""
    fi
fi

# Display summary of created resources
echo ""
echo "============================================================"
echo "RESOURCE SUMMARY"
echo "============================================================"
echo "Web ACL Name: $WEB_ACL_NAME"
echo "Web ACL ID: $WEB_ACL_ID"
echo "Web ACL ARN: $WEB_ACL_ARN"
if [ -n "${DISTRIBUTION_ID:-}" ]; then
    echo "Associated CloudFront Distribution: $DISTRIBUTION_ID"
fi
echo ""

# Auto-confirm cleanup
echo "============================================================"
echo "CLEANUP CONFIRMATION"
echo "============================================================"
echo "Proceeding with automatic cleanup of all created resources..."

echo ""
echo "============================================================"
echo "Tutorial completed!"
echo "============================================================"
echo "Log file: $LOG_FILE"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AssociateWebAcl
  - CreateWebAcl
  - DeleteWebAcl
  - DisassociateWebAcl

- GetCallerIdentity
- GetWebAcl
- ListDistributions
- UpdateWebAcl

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memindahkan rahasia hardcode untuk mengamankan penyimpanan rahasia

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM
- Buat rahasia di Secrets Manager
- Perbarui kode aplikasi Anda
- Perbarui rahasianya
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail
```

```
# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "================================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi
```

```
    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
```

```
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
```

```
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
```

```
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
```

```
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
```

```
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "========================================"
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
```

```
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Jalankan stress test CPU pada instance mesin virtual menggunakan injeksi kesalahan

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM
- Buat CloudWatch alarm
- Buat template eksperimen
- Jalankan percobaan

- Verifikasi hasilnya
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
```

```
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
```

```
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
```

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
```

```
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
```

```
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
```

```
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
```

```
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF
```

```
# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail
```

```
    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")
```

```
# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
```

```
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
 "$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"
```

```
    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData
  - GetRole
  - MonitorInstances
  - PutMetricAlarm
  - PutRolePolicy
  - RemoveRoleFromInstanceProfile
  - RunInstances
  - StartExperiment
  - StopExperiment
  - TerminateInstances
  - Tunggu

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Menyiapkan manajemen sistem

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat izin IAM untuk Systems Manager
- Membuat peran IAM untuk Systems Manager
- Konfigurasikan Systems Manager
- Verifikasi pengaturan
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori tutorial pengembang Sample.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
```

```
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "==========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
```

```
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
```

```
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
```

```
      "Effect": "Allow",
      "Action": [
        "ssm:DescribeAutomationExecutions",
        "ssm:GetAutomationExecution",
        "ssm:ListAssociations",
        "ssm:DescribeAssociation",
        "ssm:ListDocuments",
        "ssm:ListResourceDataSync",
        "ssm:DescribePatchBaselines",
        "ssm:GetPatchBaseline",
        "ssm:DescribeMaintenanceWindows",
        "ssm:DescribeMaintenanceWindowTasks"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SsmDocument",
      "Effect": "Allow",
      "Action": [
        "ssm:GetDocument",
        "ssm:DescribeDocument"
      ],
      "Resource": [
        "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
        "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
      ]
    },
    {
      "Sid": "SsmEnableExplorer",
      "Effect": "Allow",
      "Action": "ssm:StartAutomationExecution",
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
      ],
      "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
```

```
        "Sid": "OrgsReadOnly",
        "Effect": "Allow",
        "Action": [
          "organizations:DescribeAccount",
          "organizations:DescribeOrganization",
          "organizations:ListDelegatedAdministrators",
          "organizations:ListRoots",
          "organizations:ListParents",
          "organizations:ListOrganizationalUnitsForParent",
          "organizations:DescribeOrganizationalUnit",
          "organizations:ListAWSServiceAccessForOrganization"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgsAdministration",
        "Effect": "Allow",
        "Action": [
          "organizations:EnableAWSServiceAccess",
          "organizations:RegisterDelegatedAdministrator",
          "organizations:DeregisterDelegatedAdministrator"
        ],
        "Resource": "*",
        "Condition": {
          "StringEquals": {
            "organizations:ServicePrincipal": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "member.org.stacksets.cloudformation.amazonaws.com",
              "resource-explorer-2.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "CfnReadOnly",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ListStacks",
          "cloudformation:DescribeStacks",
          "cloudformation:ListStackSets",
          "cloudformation:DescribeOrganizationsAccess"
        ],
        "Resource": "*"
```

```
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
    {
      "Sid": "CfnStackSetActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
```

```
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
```

```
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
```

```
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"
```

```
# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
```

```
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
```

```
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30
```

```
# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "============================================="
echo "CREATED RESOURCES"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .

- AttachRolePolicy
- CreateConfigurationManager
- CreatePolicy
- CreateRole
- DeleteConfigurationManager
- DeletePolicy
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetConfigurationManager

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Keamanan di IAM dan AWS STS

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk AWS Identity and Access Management (IAM), lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS). Topik berikut menunjukkan cara mengkonfigurasi IAM dan AWS STS untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya IAM Anda.

Konten

- AWS kredensi keamanan
- AWS pedoman audit keamanan
- Perlindungan data di AWS Identity and Access Management
- Penebangan dan pemantauan di AWS Identity and Access Management
- Validasi kepatuhan untuk AWS Identity and Access Management
- Ketahanan di AWS Identity and Access Management
- Keamanan infrastruktur di AWS Identity and Access Management
- Konfigurasi dan analisis kerentanan di AWS Identity and Access Management

- AWS kebijakan yang dikelola untuk AWS Identity and Access Management, Penganalisis Akses IAM, dan manajer akses akun
- Fitur keamanan di luar IAM

AWS kredensi keamanan

Ketika Anda berinteraksi AWS, Anda menentukan kredensi AWS keamanan Anda untuk memverifikasi siapa Anda dan apakah Anda memiliki izin untuk mengakses sumber daya yang Anda minta. AWS menggunakan kredensi keamanan untuk mengautentikasi dan mengotorisasi permintaan Anda.

Misalnya, jika ingin mengunduh file yang dilindungi dari bucket Amazon Simple Storage Service (Amazon S3), kredensial Anda harus mengizinkan akses tersebut. Jika kredensi Anda tidak menunjukkan bahwa Anda berwenang untuk mengunduh file, tolak permintaan Anda AWS . Namun, kredensi AWS keamanan Anda tidak diperlukan bagi Anda untuk mengunduh file di bucket Amazon S3 yang dibagikan secara publik.

Ada berbagai jenis pengguna di AWS, masing-masing dengan kredensi keamanan mereka sendiri:

- Pemilik akun (pengguna root) — Pengguna yang membuat Akun AWS dan memiliki akses penuh.
- AWS IAM Identity Center pengguna — Pengguna dikelola di AWS IAM Identity Center.
- Prinsipal Federasi — Pengguna dari penyedia identitas eksternal yang diberikan akses sementara ke melalui federasi. AWS Untuk informasi lebih lanjut tentang identitas federasi, lihat. Penyedia identitas dan federasi ke AWS
- Pengguna IAM — Pengguna individu yang dibuat dalam layanan AWS Identity and Access Management (IAM).

Pengguna memiliki kredensi keamanan jangka panjang atau sementara. Pengguna root, pengguna IAM, dan kunci akses memiliki kredenal keamanan jangka panjang yang tidak kedaluwarsa. Untuk melindungi kredensi jangka panjang, ada proses untuk mengelola kunci akses, mengubah kata sandi, dan mengaktifkan MFA.

Untuk menyederhanakan pengelolaan kredensi pengguna root di seluruh akun anggota di AWS Organizations, Anda dapat mengamankan kredenal pengguna root secara terpusat dari penggunaan yang Anda kelola. Akun AWS AWS OrganizationsMengelola akses root secara terpusat untuk akun anggotamemungkinkan Anda menghapus dan mencegah pemulihan kredensi pengguna root jangka panjang secara terpusat, mencegah akses root yang tidak diinginkan dalam skala besar.

Peran IAM, pengguna di Pusat Identitas AWS IAM, dan prinsip pengguna AWS STS federasi memiliki kredenal keamanan sementara. Kredensi keamanan sementara kedaluwarsa setelah jangka waktu tertentu atau ketika pengguna mengakhiri sesi mereka. Kredensi sementara bekerja hampir identik dengan kredensi jangka panjang, dengan perbedaan berikut:

- Kredensial keamanan sementara bersifat jangka pendek, seperti namanya. Konfigurasi dapat berlangsung selama beberapa menit hingga beberapa jam. Setelah kredensialnya kedaluwarsa, AWS tidak lagi mengenalinya atau mengizinkan segala jenis akses dari permintaan API yang dibuat dengannya.
- Kredensial keamanan sementara tidak disimpan dengan pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. Ketika (atau bahkan sebelum) kredensial keamanan sementara kedaluwarsa, pengguna dapat meminta kredensial baru, selama pengguna yang memintanya masih memiliki izin untuk melakukannya.

Akibatnya, kredensi sementara memiliki keunggulan sebagai berikut dibandingkan kredensi jangka panjang:

- Anda tidak perlu mendistribusikan atau menanamkan kredensi AWS keamanan jangka panjang dengan aplikasi.
- Anda dapat memberikan akses ke AWS sumber daya Anda kepada pengguna tanpa harus menentukan AWS identitas untuk mereka. Kredensial sementara adalah dasar untuk peran dan federasi identitas.
- Kredensi keamanan sementara memiliki masa pakai yang terbatas, jadi Anda tidak perlu memperbaruinya atau mencabutnya secara eksplisit saat tidak lagi diperlukan. Setelah kredensial keamanan sementara berakhir, kredensial tersebut tidak dapat digunakan kembali. Anda dapat menentukan berapa lama kredensial berlaku, hingga batas maksimum.

Pertimbangan keamanan

Kami menyarankan Anda mempertimbangkan informasi berikut saat menentukan ketentuan keamanan untuk Anda Akun AWS:

- Saat Anda membuat Akun AWS, kami membuat pengguna root akun. Kredensi pengguna root (pemilik akun) memungkinkan akses penuh ke semua sumber daya di akun. Tugas pertama yang Anda lakukan dengan pengguna root adalah memberikan izin administratif pengguna lain kepada Anda Akun AWS sehingga Anda meminimalkan penggunaan pengguna root.

- Otentikasi multi-faktor (MFA) memberikan tingkat keamanan ekstra bagi pengguna yang dapat mengakses Anda. Akun AWS Untuk keamanan tambahan, kami sarankan Anda memerlukan MFA pada Pengguna root akun AWS kredensi dan semua pengguna IAM. Untuk informasi selengkapnya, lihat AWS Multi-factor otentikasi di IAM.
- AWS memerlukan berbagai jenis kredensi keamanan, tergantung pada bagaimana Anda mengakses AWS dan jenis AWS pengguna Anda. Misalnya, Anda menggunakan kredenal masuk untuk Konsol Manajemen AWS sementara Anda menggunakan kunci akses untuk melakukan panggilan terprogram. AWS Untuk bantuan menentukan jenis pengguna dan halaman login, lihat Apa itu AWS Masuk di AWS Sign-In Panduan Pengguna.
- Anda tidak dapat menggunakan kebijakan IAM untuk menolak akses pengguna root ke sumber daya secara eksplisit. Anda hanya dapat menggunakan kebijakan kontrol AWS Organizations layanan (SCP) untuk membatasi izin pengguna root.
- Jika Anda lupa atau kehilangan kata sandi pengguna root Anda, Anda harus memiliki akses ke alamat email yang terkait dengan akun Anda untuk mengatur ulang.
- Jika Anda kehilangan kunci akses pengguna root Anda, Anda harus dapat masuk ke akun Anda sebagai pengguna root untuk membuat yang baru.
- Jangan gunakan pengguna root untuk tugas sehari-hari Anda. Gunakan untuk melakukan tugas-tugas yang hanya dapat dilakukan oleh pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihatTugas yang memerlukan kredensial pengguna root.
- Kredensial keamanan adalah khusus akun. Jika Anda memiliki akses ke beberapa Akun AWS, Anda memiliki kredensi terpisah untuk setiap akun.
- Kebijakan menentukan tindakan apa yang dapat dilakukan pengguna, peran, atau anggota grup pengguna, pada AWS sumber daya mana, dan dalam kondisi apa. Dengan menggunakan kebijakan, Anda dapat mengontrol akses Layanan AWS dan sumber daya dengan aman di situs Anda Akun AWS. Jika Anda harus mengubah atau mencabut izin sebagai respons terhadap peristiwa keamanan, Anda menghapus atau mengubah kebijakan alih-alih membuat perubahan langsung pada identitas.
- Pastikan untuk menyimpan kredensi masuk untuk pengguna IAM Akses Darurat Anda dan kunci akses apa pun yang Anda buat untuk akses terprogram di lokasi yang aman. Jika Anda kehilangan kunci akses, Anda harus masuk ke akun Anda untuk membuat yang baru.
- Kami sangat menyarankan agar Anda menggunakan kredenal sementara yang disediakan oleh peran IAM dan prinsipal federasi alih-alih kredensi jangka panjang yang disediakan oleh pengguna IAM dan kunci akses.

Akses terprogram dengan AWS kredensial keamanan

Sebaiknya gunakan tombol akses jangka pendek bila memungkinkan untuk melakukan panggilan terprogram ke AWS atau menggunakan AWS Command Line Interface atau Alat AWS untuk PowerShell. Namun, Anda juga dapat menggunakan kunci AWS akses jangka panjang untuk tujuan ini.

Saat Anda membuat kunci akses jangka panjang, Anda membuat ID kunci akses (misalnya,`AKIAIOSFODNN7EXAMPLE`) dan kunci akses rahasia (misalnya,`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) sebagai satu set. Secret access key hanya tersedia untuk diunduh saat Anda membuatnya. Jika tidak mengunduh secret access key atau kehilangannya, Anda harus membuat yang baru.

Dalam banyak skenario, Anda tidak memerlukan kunci akses jangka panjang yang tidak pernah kedaluwarsa (seperti yang Anda miliki saat membuat kunci akses untuk pengguna IAM). Sebagai gantinya, Anda dapat membuat IAM role dan membuat kredensial keamanan sementara. KredenSIAL keamanan sementara termasuk ID kunci akses dan kunci akses rahasia, tetapi mereka juga menyertakan token keamanan yang menunjukkan kapan kredenSIAL kedaluwarsa. Setelah kedaluwarsa, mereka tidak lagi valid. Untuk informasi selengkapnya, lihat Alternatif untuk kunci akses jangka panjang

ID kunci akses dimulai dengan `AKIA` adalah kunci akses jangka panjang untuk pengguna IAM atau pengguna Akun AWS root. ID kunci akses yang dimulai dengan `ASIA` adalah kunci akses kredensional sementara yang Anda buat menggunakan AWS STS operasi.

Pengguna memerlukan akses terprogram jika mereka ingin berinteraksi dengan AWS di luar. Konsol Manajemen AWS Cara untuk memberikan akses terprogram tergantung pada jenis pengguna yang mengakses AWS.

Untuk memberi pengguna akses programatis, pilih salah satu opsi berikut.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
IAM	(Disarankan) Gunakan kredenSIAL konsol sebagai kredenSIAL sementara untuk menandatangani permintaan	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
	terprogram ke AWS CLI, AWS SDK, atau AWS API.	• Untuk AWS CLI, lihat Login untuk pengembangan AWS lokal di Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK, lihat Login untuk pengembangan AWS lokal di Panduan Refer AWS ensi SDK dan Alat.
Identitas tenaga kerja (Pengguna yang dikelola di Pusat Identitas IAM)	Gunakan kredenSIAL sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau AWS API.	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk AWS CLI, lihat Meng onfigurasi AWS CLI untuk digunakan AWS IAM Identity Center di Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK, alat, dan AWS API, lihat otentikasi Pusat Identitas IAM di Panduan Refer AWS ensi SDK dan Alat.
IAM	Gunakan kredenSIAL sementara untuk menandatangani permintaan terprogram ke AWS CLI, AWS SDK, atau AWS API.	Mengikuti petunjuk dalam Menggunakan kredenSIAL sementara dengan AWS sumber daya di Panduan Pengguna IAM.

Pengguna mana yang membutuhkan akses programatis?	Untuk	Oleh
IAM	(Tidak direkomendasikan) Gunakan kredentif jangka panjang untuk menandata ngani permintaan terprogram ke AWS CLI, AWS SDK, atau AWS API.	Mengikuti petunjuk untuk antarmuka yang ingin Anda gunakan. • Untuk itu AWS CLI, lihat Meng autentikasi menggunakan kredentif pengguna IAM di Panduan AWS Command Line Interface Pengguna. • Untuk AWS SDK dan alat, lihat Meng autentikasi menggunakan kredentif jangka panjang di Panduan Refer AWS ensi SDK dan Alat. • Untuk AWS API, lihat M engelola kunci akses untuk pengguna IAM di Panduan Pengguna IAM.

Alternatif untuk kunci akses jangka panjang

Untuk banyak kasus penggunaan umum, ada alternatif untuk kunci akses jangka panjang. Untuk meningkatkan keamanan akun Anda, pertimbangkan hal berikut.

- Jangan menyematkan kunci akses jangka panjang dan kunci akses rahasia dalam kode aplikasi Anda atau di repositori kode — Sebagai gantinya, gunakan AWS Secrets Manager, atau solusi manajemen rahasia lainnya, sehingga Anda tidak perlu melakukan hardcode kunci dalam teks biasa. Aplikasi atau klien kemudian dapat mengambil rahasia bila diperlukan. Untuk informasi lebih lanjut, lihat Apa itu AWS Secrets Manager? dalam Panduan AWS Secrets Manager Pengguna.

- Gunakan peran IAM untuk menghasilkan kredenSIAL keamanan sementara bila memungkinkan — Sel alu gunakan mekanisme untuk mengeluarkan kredenSIAL keamanan sementara bila memungkinkan, bukan kunci akses jangka panjang. KredenSIAL keamanan sementara lebih aman karena tidak disimpan bersama pengguna tetapi dihasilkan secara dinamis dan diberikan kepada pengguna saat diminta. KredenSIAL keamanan sementara memiliki masa pakai terbatas sehingga Anda tidak perlu mengelola atau memperbaruinya. Mekanisme yang menyediakan kunci akses sementara termasuk peran IAM atau otentikasi pengguna Pusat Identitas IAM. Untuk mesin yang berjalan di luar AWS Anda dapat menggunakan AWS Identity and Access Management Roles Anywhere.
- Gunakan alternatif untuk kunci akses jangka panjang untuk AWS Command Line Interface (AWS CLI) atau **`aws-shell`** — Alternatif termasuk yang berikut ini.
  - Login untuk pengembangan AWS lokal dengan kredenSIAL konsol. Anda dapat menggunakan AWS CLI Versi 2 dan `aws login` perintah untuk menghasilkan kredenSIAL jangka pendek untuk menjalankan AWS CLI perintah menggunakan kredenSIAL konsol Anda. Untuk mempelajari lebih lanjut, lihat Login untuk pengembangan AWS lokal di Panduan AWS Command Line Interface Pengguna.
  - AWS CloudShelladalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari. Konsol Manajemen AWS Anda dapat menjalankan AWS CLI perintah melawan Layanan AWS melalui shell pilihan Anda (Bash, Powershell, atau Z shell). Ketika Anda melakukan ini, Anda tidak perlu mengunduh atau menginstal alat baris perintah. Untuk informasi selengkapnya, lihat Apa itu AWS CloudShell? dalam Panduan Pengguna AWS CloudShell .
- Jangan membuat kunci akses jangka panjang untuk pengguna manusia yang memerlukan akses ke aplikasi atau Layanan AWS — Pusat Identitas IAM dapat menghasilkan kredentif akses sementara untuk diakses oleh pengguna IdP eksternal Anda. Layanan AWS Ini menghilangkan kebutuhan untuk membuat dan mengelola kredenSIAL jangka panjang di IAM. Di Pusat Identitas IAM, buat kumpulan izin Pusat Identitas IAM yang memberikan akses pengguna IdP eksternal. Kemudian tetapkan grup dari Pusat Identitas IAM ke kumpulan izin di yang dipilih Akun AWS. Untuk informasi selengkapnya, lihat Apa AWS IAM Identity Center itu, Menghubungkan ke penyedia identitas eksternal Anda, dan Kumpulan izin di Panduan AWS IAM Identity Center Pengguna.
- Jangan menyimpan kunci akses jangka panjang dalam layanan AWS komputasi — Sebagai gantinya, tetapkan peran IAM untuk menghitung sumber daya. Ini secara otomatis memasok kredenSIAL sementara untuk memberikan akses. Misalnya, saat Anda membuat profil instans yang dilampirkan ke instans Amazon EC2, Anda dapat menetapkan AWS peran ke instans dan membuatnya tersedia untuk semua aplikasinya. Profil instance berisi peran dan memungkinkan

program yang berjalan di instans Amazon EC2 untuk mendapatkan kredenSIAL sementara. Untuk mempelajari selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2.

AWS pedoman audit keamanan

Audit konfigurasi keamanan Anda secara berkala untuk memastikannya memenuhi kebutuhan bisnis Anda saat ini. Audit memberi Anda kesempatan untuk menghapus pengguna, peran, grup, dan kebijakan IAM yang tidak dibutuhkan, dan untuk memastikan bahwa pengguna dan perangkat lunak Anda tidak memiliki izin yang berlebihan.

Berikut ini adalah panduan untuk meninjau dan memantau AWS sumber daya Anda secara sistematis untuk praktik terbaik keamanan.

> **Tip**
>
> Anda dapat memantau penggunaan IAM karena berkaitan dengan praktik terbaik keamanan dengan menggunakan. AWS Security Hub CSPM Security Hub CSPM menggunakan kontrol keamanan untuk mengevaluasi konfigurasi sumber daya dan standar keamanan untuk membantu Anda mematuhi berbagai kerangka kerja kepatuhan. Untuk informasi selengkapnya tentang penggunaan CSPM Security Hub untuk mengevaluasi sumber daya IAM, lihat kontrol AWS Identity and Access Management di Panduan Pengguna. AWS Security Hub CSPM

Daftar Isi

- Kapan melakukan audit keamanan
- Pedoman untuk audit
- Tinjau kredensi AWS akun Anda
- Tinjau pengguna IAM Anda
- Tinjau grup IAM Anda
- Tinjau IAM role Anda
- Tinjau penyedia IAM untuk SAML and OpenID Connect (OIDC) Anda
- Tinjau aplikasi seluler Anda
- Tips untuk meninjau kebijakan IAM

Kapan melakukan audit keamanan

Audit konfigurasi keamanan Anda dalam situasi berikut:

- Secara periodik. Lakukan langkah-langkah yang dijelaskan dalam dokumen ini secara berkala sebagai praktik terbaik untuk keamanan.
- Jika ada perubahan di organisasi Anda, seperti orang yang keluar.
- Jika Anda telah berhenti menggunakan satu atau beberapa AWS layanan individual untuk memverifikasi bahwa Anda telah menghapus izin yang tidak lagi diperlukan oleh pengguna di akun Anda.
- Jika Anda telah menambahkan atau menghapus perangkat lunak di akun Anda, seperti aplikasi di instans Amazon EC2, OpsWorks tumpukan, templat, dll. CloudFormation
- Jika Anda mencurigai bahwa orang yang tidak berwenang mungkin telah mengakses akun Anda.

Pedoman untuk audit

Saat meninjau konfigurasi keamanan akun Anda, ikuti panduan berikut:

- Jadilah teliti. Lihatlah semua aspek konfigurasi keamanan Anda, termasuk yang jarang digunakan.
- Jangan berasumsi. Jika Anda tidak terbiasa dengan beberapa aspek konfigurasi keamanan Anda (misalnya, alasan di balik kebijakan tertentu atau keberadaan peran), selidiki kebutuhan bisnis sampai Anda memahami potensi risikonya.
- Tetap sederhana. Untuk mempermudah audit (dan manajemen), gunakan grup IAM, peran IAM, skema penamaan yang konsisten, dan kebijakan langsung.

Tinjau kredensi AWS akun Anda

Ambil langkah-langkah ini saat Anda mengaudit kredensi AWS akun Anda:

1. Jika Anda memiliki kunci akses untuk pengguna root yang tidak Anda gunakan, Anda dapat menghapusnya. Kami sangat menyarankan agar Anda tidak menggunakan kunci akses root untuk pekerjaan sehari-hari AWS, dan sebaliknya Anda menggunakan pengguna dengan kredensi sementara, seperti itu. pengguna di Pusat Identitas AWS IAM
2. Jika Anda memerlukan kunci akses untuk akun Anda, pastikan Anda memperbaruinya saat diperlukan.

Tinjau pengguna IAM Anda

Lakukan langkah-langkah berikut saat Anda mengaudit pengguna IAM yang ada:

1. Buat daftar pengguna Anda dan kemudian hapus pengguna yang tidak diperlukan.
2. Hapus pengguna dari grup yang tidak mereka perlukan aksesnya.
3. Tinjau kebijakan yang dilampirkan ke grup tempat pengguna berada. Lihat Tips untuk meninjau kebijakan IAM.
4. Hapus kredensial keamanan yang tidak dibutuhkan pengguna atau yang mungkin telah diekspos. Misalnya, pengguna IAM yang digunakan untuk aplikasi tidak memerlukan kata sandi (yang diperlukan hanya untuk masuk ke AWS situs web). Demikian pula, jika pengguna tidak menggunakan kunci akses, tidak ada alasan bagi pengguna untuk memilikinya. Untuk informasi selengkapnya, lihat Mengelola Kata Sandi untuk pengguna IAM dan Mengelola Kunci Akses untuk pengguna IAM.

   Anda dapat membuat dan mengunduh laporan kredensial yang mencantumkan semua pengguna IAM di akun Anda dan status berbagai kredensial mereka, termasuk kata sandi, access key, dan perangkat MFA. Untuk kata sandi dan kunci akses, laporan kredensi menunjukkan tanggal dan waktu ketika kata sandi atau kunci akses terakhir digunakan. Pertimbangkan untuk menghapus kredensil yang belum digunakan baru-baru ini dari akun Anda. (Jangan hapus pengguna Akses Darurat Anda.) Untuk informasi selengkapnya, lihat Mendapatkan Laporan Kredensi untuk AWS Akun Anda.
5. Perbarui kata sandi dan kunci akses bila diperlukan untuk kasus penggunaan yang memerlukan kredensi jangka panjang. Untuk informasi selengkapnya, lihat Mengelola Kata Sandi untuk Pengguna IAM dan Mengelola Kunci Akses untuk pengguna IAM.
6. Sebagai praktik terbaik, mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara. Jika memungkinkan, transisi dari pengguna IAM ke prinsipal federasi, seperti pengguna di IAM Identity Center. Pertahankan jumlah minimum pengguna IAM yang dibutuhkan untuk aplikasi Anda.

Tinjau grup IAM Anda

Lakukan langkah-langkah berikut saat Anda mengaudit grup IAM:

1. Buat daftar grup Anda, lalu hapus grup yang tidak Anda gunakan.
2. Tinjau pengguna di setiap grup dan hapus pengguna yang tidak termasuk.

3. Tinjau kebijakan yang dilampirkan ke grup. Lihat Tips untuk meninjau kebijakan IAM.

Tinjau IAM role Anda

Lakukan langkah-langkah berikut saat Anda mengaudit IAM role:

1. Buat daftar peran Anda, lalu hapus peran yang tidak Anda gunakan.
2. Tinjau kebijakan kepercayaan peran. Pastikan Anda mengetahui siapa akun utamanya dan memahami mengapa akun atau pengguna tersebut harus dapat mengambil peran tersebut.
3. Tinjau kebijakan akses untuk peran tersebut guna memastikan bahwa kebijakan akses memberikan izin yang sesuai kepada siapa pun yang mengambil peran tersebut—lihat Tips untuk meninjau kebijakan IAM.

Tinjau penyedia IAM untuk SAML and OpenID Connect (OIDC) Anda

Jika Anda telah membuat entitas IAM untuk membangun kepercayaan dengan penyedia identitas SAMP atau OIDC (IDP), lakukan langkah-langkah berikut:

1. Hapus penyedia yang tidak digunakan.
2. Unduh dan tinjau dokumen AWS metadata untuk setiap IDP SAMP dan pastikan dokumen tersebut mencerminkan kebutuhan bisnis Anda saat ini.
3. Dapatkan dokumen metadata terbaru dari SAMP IdPs dan perbarui penyedia di IAM.

Tinjau aplikasi seluler Anda

Jika Anda telah membuat aplikasi seluler yang membuat permintaan AWS, lakukan langkah-langkah berikut:

1. Pastikan aplikasi seluler tidak berisi kunci akses yang disematkan, meskipun berada di penyimpanan terenkripsi.
2. Dapatkan kredensi sementara untuk aplikasi dengan menggunakan yang APIs dirancang untuk tujuan itu.

> **Note**
>
> Sebaiknya gunakan Amazon Cognito untuk mengelola identitas pengguna di aplikasi Anda. Layanan ini memungkinkan Anda mengautentikasi pengguna menggunakan Login with Amazon, Facebook, Google, atau penyedia identitas yang kompatibel dengan OpenID Connect (OIDC). Untuk informasi selengkapnya, lihat kumpulan identitas Amazon Cognito di Panduan Pengembang Amazon Cognito.

Tips untuk meninjau kebijakan IAM

Kebijakan sangat kuat dan halus, jadi penting untuk mempelajari dan memahami izin yang diberikan oleh setiap kebijakan. Gunakan pedoman berikut saat meninjau kebijakan:

- Lampirkan kebijakan ke grup atau peran, bukan ke pengguna individu. Jika pengguna individu memiliki kebijakan, pastikan Anda memahami mengapa pengguna tersebut memerlukan kebijakan tersebut.
- Pastikan bahwa pengguna, grup, dan peran IAM memiliki izin yang mereka butuhkan dan tidak memiliki izin tambahan.
- Gunakan IAM Policy Simulator untuk menguji kebijakan yang dilampirkan ke pengguna atau grup.
- Ingatlah bahwa izin pengguna adalah hasil dari semua kebijakan yang berlaku — baik kebijakan berbasis identitas (pada pengguna, grup, atau peran) maupun kebijakan berbasis sumber daya (pada sumber daya seperti bucket Amazon S3, antrian Amazon SQS, topik Amazon SNS, dan kunci Amazon SNS). AWS KMS Penting untuk memeriksa semua kebijakan yang berlaku bagi pengguna dan untuk memahami set lengkap izin yang diberikan kepada pengguna individu.
- Ketahuilah bahwa mengizinkan pengguna untuk membuat pengguna, grup, peran, atau kebijakan IAM dan melampirkan kebijakan ke entitas utama secara efektif memberikan pengguna tersebut semua izin ke semua sumber daya di akun Anda. Pengguna yang dapat membuat kebijakan dan melampirkannya ke pengguna, grup, atau peran dapat memberikan izin apa pun kepada diri mereka sendiri. Secara umum, jangan berikan izin IAM kepada pengguna atau peran yang tidak Anda percayai dengan akses penuh ke sumber daya di akun Anda. Saat melakukan audit keamanan, konfirmasikan bahwa izin IAM berikut diberikan kepada identitas tepercaya:
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`

  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Pastikan kebijakan tidak memberikan izin untuk layanan yang tidak Anda gunakan. Misalnya, jika Anda menggunakan kebijakan AWS terkelola, pastikan kebijakan AWS terkelola yang digunakan di akun Anda adalah untuk layanan yang benar-benar Anda gunakan. Untuk mengetahui kebijakan AWS terkelola mana yang digunakan di akun Anda, gunakan `GetAccountAuthorizationDetails`API IAM (AWS CLI perintah: `aws iam get-account-authorization-details`).
- Jika kebijakan memberikan izin kepada pengguna untuk meluncurkan instans Amazon EC2, kebijakan tersebut mungkin juga mengizinkan tindakan `iam:PassRole` tersebut, tetapi jika demikian, kebijakan tersebut harus secara eksplisit mencantumkan peran yang dapat diteruskan pengguna ke instans Amazon EC2.
- Periksa setiap nilai untuk `Action` atau `Resource` elemen yang termasuk`*`. Jika memungkinkan, berikan `Allow` akses ke tindakan individu dan sumber daya yang dibutuhkan pengguna. Namun, berikut ini adalah alasan yang mungkin cocok untuk menggunakan `*` dalam kebijakan:
  - Kebijakan ini dirancang untuk memberikan izin tingkat administratif.
  - Karakter wildcard digunakan untuk set tindakan serupa (misalnya, `Describe*`) sebagai kenyamanan dan Anda merasa nyaman dengan daftar lengkap tindakan yang direferensikan dengan cara ini.
  - Karakter wildcard digunakan untuk menunjukkan kelas sumber daya atau jalur sumber daya (misalnya,`arn:aws:iam::`*`account-id`*`:users/division_abc/*`), dan Anda merasa nyaman memberikan akses ke semua sumber daya di kelas atau jalur itu.
  - Tindakan layanan tidak mendukung izin tingkat sumber daya, dan satu-satunya pilihan untuk sumber daya adalah. `*`
- Periksa nama kebijakan untuk memastikan mereka mencerminkan fungsi kebijakan. Misalnya, meskipun kebijakan mungkin memiliki nama yang menyertakan "hanya baca", kebijakan tersebut mungkin benar-benar memberikan izin tulis atau ubah.

Untuk informasi selengkapnya tentang perencanaan audit keamanan Anda, lihat Praktik Terbaik untuk Keamanan, Identitas, dan Kepatuhan di Pusat AWS Arsitektur.

Perlindungan data di AWS Identity and Access Management

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS Identity and Access Management. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat FAQ Privasi Data AWS . Untuk informasi tentang perlindungan data di Eropa, lihat Pusat Peraturan Perlindungan Data Umum (GDPR).

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensyal dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan IAM atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDK. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log

diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Enkripsi data dalam IAM dan AWS STS

Enkripsi data biasanya terbagi dalam dua kategori: enkripsi saat diam dan enkripsi dalam transit.

Enkripsi saat diam

Data yang dikumpulkan dan disimpan oleh IAM dienkripsi saat diam.

- IAM — Data yang dikumpulkan dan disimpan dalam IAM mencakup alamat IP, metadata akun pelanggan, dan data identifikasi pelanggan yang menyertakan kata sandi. Metadata akun pelanggan dan data pengenal pelanggan dienkripsi saat diam menggunakan AES 256 atau di-hash menggunakan SHA 256.
- AWS STS— AWS STS tidak mengumpulkan konten pelanggan kecuali untuk log layanan yang mencatat permintaan yang berhasil, salah, dan salah ke layanan.

Enkripsi saat bergerak

Data identifikasi pelanggan, termasuk kata sandi, dienkripsi dalam perjalanan menggunakan TLS 1.2 dan 1.3. Semua AWS STS titik akhir mendukung HTTPS untuk mengenkripsi data dalam perjalanan. Untuk daftar AWS STS titik akhir, lihatAWS STS Wilayah dan titik akhir.

Manajemen kunci dalam IAM dan AWS STS

Anda tidak dapat mengelola kunci enkripsi menggunakan IAM atau AWS STS. Untuk informasi selengkapnya tentang kunci enkripsi, lihat Apa itu AWS KMS? di Panduan AWS Key Management Service Pengembang

Privasi lalu lintas internetwork di IAM dan AWS STS

Permintaan untuk IAM harus dilakukan menggunakan protokol Transport Layer Security (TLS). Anda dapat mengamankan koneksi ke AWS STS layanan dengan menggunakan titik akhir VPC. Untuk mempelajari informasi lebih lanjut, lihat Titik akhir VPC antarmuka.

Penebangan dan pemantauan di AWS Identity and Access Management

Pemantauan adalah bagian penting dari menjaga keandalan, ketersediaan, dan kinerja AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS) dan AWS solusi Anda yang lain. AWS menyediakan beberapa alat untuk memantau AWS sumber daya Anda dan menanggapi insiden potensial:

- AWS CloudTrailmenangkap semua panggilan API untuk IAM dan AWS STS sebagai peristiwa, termasuk panggilan dari konsol dan panggilan API. Untuk mempelajari lebih lanjut tentang menggunakan CloudTrail dengan IAM dan AWS STS, lihatPencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail. Untuk informasi selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.
- AWS Identity and Access Management Access Analyzer membantu Anda mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket Amazon S3 atau peran IAM, yang dibagikan dengan entitas eksternal. Hal ini membantu Anda mengidentifikasi akses yang tidak diinginkan ke sumber daya dan data Anda, yang merupakan risiko keamanan. Untuk mempelajari lebih lanjut, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?
- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang ditentukan. Misalnya, Anda dapat CloudWatch melacak penggunaan CPU atau metrik lain dari EC2 instans Amazon Anda dan secara otomatis meluncurkan instans baru bila diperlukan. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.
- Amazon CloudWatch Logs membantu Anda memantau, menyimpan, dan mengakses file log Anda dari EC2 instans Amazon CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log dalam penyimpanan yang sangat durabel. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon CloudWatch Logs.

Untuk sumber daya tambahan dan praktik keamanan terbaik untuk IAM, lihat Praktik terbaik keamanan dan kasus penggunaan di AWS Identity and Access Management.

Topik

- Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail
- Lacak tugas istimewa di AWS CloudTrail

Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail

IAM dan AWS STS terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna atau peran IAM. CloudTrail menangkap semua panggilan API untuk IAM dan acara AWS STS as, termasuk panggilan dari konsol dan dari panggilan API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon S3. Jika Anda tidak membuat konfigurasi jejak, Anda masih dapat melihat kejadian terbaru dalam konsol CloudTrail di Riwayat peristiwa. Anda dapat menggunakan CloudTrail untuk mendapatkan informasi tentang permintaan yang dibuat ke IAM atau AWS STS. Misalnya, Anda dapat melihat alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan itu dibuat, dan rincian tambahan.

Untuk mempelajari selengkap CloudTrail nya, lihat Panduan AWS CloudTrail Pengguna.

Topik

- IAM dan AWS STS informasi di CloudTrail
- Pencatatan IAM dan AWS STS Permintaan API
- Pencatatan permintaan API ke yang lain AWS layanan
- Mencatat peristiwa masuk pengguna
- Mencatat peristiwa masuk untuk kredensial sementara
- Contoh peristiwa IAM API di log CloudTrail
- Contoh AWS STS Peristiwa API dalam CloudTrail log
- Contoh peristiwa masuk di log CloudTrail
- Perilaku kebijakan kepercayaan peran IAM

IAM dan AWS STS informasi di CloudTrail

CloudTrail diaktifkan pada Anda Akun AWS saat Anda membuat akun. Ketika aktivitas terjadi di IAM atau AWS STS, aktivitas tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya di riwayat acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di Akun AWS. Untuk informasi selengkapnya, lihat Melihat Acara dengan Ri CloudTrail wayat Acara.

Untuk catatan acara yang berkelanjutan di Anda Akun AWS, termasuk acara untuk IAM dan AWS STS, buat jejak. Jejak CloudTrail memungkinkan pengiriman file log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol tersebut, jejak diterapkan ke semua Wilayah. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan bertindak berdasarkan data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengonfigurasi Pemberitahuan Amazon SNS untuk CloudTrail
- Mener CloudTrail ima File Log dari Beberapa Wilayah dan M CloudTrail enerima File Log dari Beberapa Akun

Semua IAM dan AWS STS tindakan dicatat oleh CloudTrail dan didokumentasikan dalam Referensi API IAM dan Referensi AWS Security Token Service API.

Pencatatan IAM dan AWS STS Permintaan API

CloudTrail mencatat semua permintaan API yang diautentikasi ke operasi IAM dan AWS STS API. CloudTrail juga mencatat permintaan yang tidak diautentikasi ke AWS STS tindakan, `AssumeRoleWithSAML` dan`AssumeRoleWithWebIdentity`, dan mencatat informasi yang disediakan oleh penyedia identitas. Namun, beberapa AWS STS permintaan yang tidak diautentikasi mungkin tidak dicatat karena tidak memenuhi harapan minimum untuk cukup valid untuk dipercaya sebagai permintaan yang sah. CloudTrail tidak mencatat AWS STS permintaan yang ditolak di akun target untuk asumsi peran lintas akun dan sesi istimewa yang dimulai melalui `AssumeRoot` API.

Anda dapat menggunakan informasi yang dicatat untuk memetakan panggilan yang dilakukan oleh prinsipal federasi OIDC atau SAML dengan peran yang diasumsikan kembali ke pemanggil federasi eksternal yang berasal. Dalam kasus`AssumeRole`, Anda dapat memetakan panggilan kembali ke AWS layanan asal atau ke akun pengguna asal. `userIdentity`Bagian data JSON dalam entri CloudTrail log berisi informasi yang Anda perlukan untuk memetakan AssumeRole* permintaan dengan prinsip sesi tertentu. Untuk informasi selengkapnya, lihat CloudTrail UserIdentity Element di Panduan AWS CloudTrail Pengguna.

AWS CloudTrail log akan berisi informasi MFA saat pengguna IAM masuk dengan MFA. Jika pengguna IAM mengambil peran IAM, juga CloudTrail akan `mfaAuthenticated: true` masuk

`sessionContext` atribut untuk tindakan yang dilakukan menggunakan peran yang diasumsikan. Namun, CloudTrail logging terpisah dari apa yang diperlukan IAM ketika panggilan API dilakukan dengan kredentif peran yang diasumsikan. Untuk informasi selengkapnya, lihat CloudTrail UserIdentity Element.

Misalnya, panggilan ke IAM`CreateUser`,, `DeleteRoleListGroups`, dan operasi API lainnya semuanya dicatat oleh CloudTrail.

Contoh untuk jenis entri log ini akan disampaikan nanti dalam topik ini.

Pencatatan permintaan API ke yang lain AWS layanan

Permintaan yang diautentikasi ke operasi API AWS layanan lainnya dicatat oleh CloudTrail, dan entri log ini berisi informasi tentang siapa yang membuat permintaan.

Misalnya, anggap bahwa Anda mengajukan permintaan untuk mendaftar instans Amazon EC2 atau membuat grup penyebaran AWS CodeDeploy . Perincian tentang orang atau layanan yang mengajukan permintaan tersebut terdapat di entri log untuk permintaan tersebut. Informasi ini membantu Anda menentukan apakah permintaan dibuat oleh Pengguna root akun AWS, pengguna IAM, peran, atau AWS layanan lain.

Untuk detail selengkapnya tentang informasi identitas pengguna dalam entri CloudTrail log, lihat Elem en Identitas Pengguna di Panduan AWS CloudTrail Pengguna.

Mencatat peristiwa masuk pengguna

CloudTrail mencatat peristiwa masuk ke Konsol Manajemen AWS, alat pengembangan lokal seperti AWS CLI dan SDK, forum AWS diskusi, dan. AWS Marketplace CloudTrailmencatat upaya masuk yang berhasil dan gagal untuk pengguna IAM, prinsipal federasi SAML dan OIDC, dan prinsipal pengguna federasi. AWS STS

Untuk melihat contoh CloudTrail peristiwa untuk login pengguna root yang berhasil dan tidak berhasil, lihat Contoh catatan peristiwa untuk pengguna root di Panduan AWS CloudTrail Pengguna.

Sebagai praktik terbaik keamanan, AWS tidak mencatat teks nama pengguna IAM yang dimasukkan saat kegagalan masuk disebabkan oleh nama pengguna yang salah. Teks nama pengguna ditutupi oleh nilai `HIDDEN_DUE_TO_SECURITY_REASONS`. Untuk contoh ini, lihat Contoh peristiwa gagal masuk yang disebabkan oleh nama pengguna yang salah, nanti dalam topik ini. Teks nama pengguna disamarkan karena kegagalan tersebut mungkin disebabkan oleh kesalahan pengguna.

Pembuatan catatan kesalahan ini dapat memaparkan informasi yang berpotensi sensitif. Sebagai contoh:

- Anda secara tidak sengaja memasukkan kata sandi anda di kotak nama pengguna.
- Anda memilih tautan untuk halaman masuk salah satu Akun AWS, tetapi kemudian mengetikkan nomor akun untuk yang lain. Akun AWS
- Anda lupa akun mana yang sedang Anda masuki dan secara tidak sengaja mengetikkan nama akun dari akun email pribadi Anda, pengenal masuk bank Anda, atau beberapa ID pribadi lainnya.

Mencatat peristiwa masuk untuk kredensial sementara

Ketika seorang pelaku utama meminta kredensial sementara, jenis utama menentukan bagaimana CloudTrail mencatat peristiwa. Ini bisa jadi rumit ketika pelaku utama mengambil peran dalam akun lain. Terdapat beberapa panggilan API untuk melakukan operasi terkait operasi lintas akun peran. Pertama, kepala sekolah memanggil AWS STS API untuk mengambil kredenSIAL sementara. Operasi itu dicatat di akun panggilan dan akun tempat AWS STS operasi dilakukan. Kemudian pelaku utama menggunakan peran tersebut untuk melakukan panggilan API lainnya dalam akun peran yang diasumsikan.

Anda dapat menggunakan kunci syarat `sts:SourceIdentity` dalam peran kebijakan kepercayaan untuk mengharuskan pengguna menentukan nama sesi saat mereka mengasumsikan sebuah peran. Misalnya, Anda dapat meminta pengguna IAM menentukan nama pengguna mereka sendiri sebagai identitas sumber mereka. Ini dapat membantu Anda menentukan pengguna mana yang melakukan tindakan tertentu di AWS. Untuk informasi selengkapnya, lihat `sts:SourceIdentity`. Anda dapat menggunakan `sts:RoleSessionName` untuk mengharuskan pengguna menentukan nama sesi saat mereka mengasumsikan sebuah peran. Ini dapat membantu Anda membedakan antara sesi peran untuk peran yang digunakan oleh prinsipal yang berbeda saat Anda meninjau AWS CloudTrail log.

Tabel berikut menunjukkan cara mencatat CloudTrail informasi identitas pengguna yang berbeda untuk setiap AWS STS API yang menghasilkan kredenSIAL sementara.

Tipe utama	API STS	Identitas pengguna di CloudTrail log untuk akun penelepon	Identitas pengguna di CloudTrail log untuk akun peran yang diasumsikan	Identitas pengguna dalam CloudTrail log untuk panggilan API peran berikutnya
Pengguna root akun AWS kredensibilitas	GetSessionToken	Identitas pengguna root	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna root
Pengguna root akun AWS kredensibilitas	AssumeRoot	Sesi pengguna root	Nomor akun dan ID utama (jika pengguna)	Sesi pengguna root
Pengguna IAM	GetSessionToken	Identitas pengguna IAM	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna IAM
Pengguna IAM	GetFederationToken	Identitas pengguna IAM	Akun pemilik peran sama dengan akun panggilan	Identitas pengguna IAM
Pengguna IAM	AssumeRole	Identitas pengguna IAM	Nomor akun dan ID utama (jika pengguna), atau prinsi AWS pal layanan	Khusus identitas peran (tanpa pengguna)
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithSAML	n/a	Identitas pengguna SAML	Khusus identitas peran (tanpa pengguna)

Tipe utama	API STS	Identitas pengguna di CloudTrail log untuk akun penelepon	Identitas pengguna di CloudTrail log untuk akun peran yang diasumsikan	Identitas pengguna dalam CloudTrail log untuk panggilan API peran berikutnya
Pengguna yang diautentikasi secara eksternal	AssumeRoleWithWebIdentity	n/a	OIDC/Web identitas pengguna	Khusus identitas peran (tanpa pengguna)

CloudTrail mempertimbangkan tindakan hanya baca jika tidak memiliki efek mutasi pada sumber daya. Saat mencatat peristiwa read-only, menyun CloudTrail ting `responseElements` informasi di log. Saat CloudTrail mencatat peristiwa yang tidak hanya dibaca, lengkapnya `responseElements` ditampilkan di entri log. Untuk AWS STS API`AssumeRole`,`AssumeRoleWithSAML`, dan`AssumeRoleWithWebIdentity`, meskipun dicatat sebagai read-only, CloudTrail akan menyertakan lengkapnya `responseElements` kecuali `secretAccessKey` dalam log untuk API ini.

Tabel berikut menunjukkan bagaimana CloudTrail log `responseElements` dan `readOnly` informasi untuk masing-masing AWS STS API yang menghasilkan kredenSIAL sementara.

API STS	Informasi elemen respons	Read-only
AssumeRole	Termasuk	true
AssumeRoleWithSAML	Termasuk	true
AssumeRoleWithWebIdentity	Termasuk	true
AssumeRoot	Termasuk	false
GetFederationToken	Termasuk	false
GetSessionToken	Termasuk	false

Contoh peristiwa IAM API di log CloudTrail

CloudTrail file log berisi peristiwa yang diformat menggunakan JSON. Sebuah peristiwa API mewakili satu permintaan API dan mencakup informasi tentang prinsipal, tindakan yang diminta, parameter apapun, dan tanggal dan waktu tindakan.

Contoh peristiwa IAM API dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk `GetUserPolicy` tindakan IAM.


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.16.96 Python/2.7.8 Linux/10 botocore/1.12.86",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePoliccyName"
    },
```

```
    "responseElements": null,
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    }
}
```

Dari informasi peristiwa ini, Anda dapat menentukan bahwa permintaan dibuat untuk mendapatkan kebijakan pengguna bernama `ReadOnlyAccess-JaneDoe-201407151307` untuk pengguna `JaneDoe`, sebagaimana ditentukan dalam elemen `requestParameters`. Anda juga dapat melihat bahwa permintaan tersebut dibuat oleh pengguna IAM bernama `JaneDoe` pada tanggal 15 Juli 2014 pada pukul 21.40 (UTC). Dalam hal ini, permintaan berasal dari Konsol Manajemen AWS, seperti yang dapat Anda ketahui dari `userAgent` elemen.

Contoh AWS STS Peristiwa API dalam CloudTrail log

CloudTrail file log berisi peristiwa yang diformat menggunakan JSON. Sebuah peristiwa API mewakili satu permintaan API dan mencakup informasi tentang prinsipal, tindakan yang diminta, parameter apapun, dan tanggal dan waktu tindakan.

Contoh lintas akun AWS STS Peristiwa API dalam file CloudTrail log

Pengguna IAM yang diberi nama `John` di akun 777788889999 memanggil AWS STS AssumeRole tindakan untuk mengambil peran di akun 111122223333. `EC2-dev` Administrator akun mengharuskan pengguna untuk menetapkan identitas sumber yang sama dengan nama pengguna mereka ketika mengambil peran. Pengguna melewati nilai identitas sumber `John`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/John",
    "accountId": "777788889999",
```

```
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "John"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
    },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Contoh kedua menunjukkan entri CloudTrail log akun peran yang diasumsikan (111122223333) untuk permintaan yang sama.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
      },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Contoh AWS STS peristiwa API rantai peran dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat oleh John Doe di akun 111111111111. John sebelumnya menggunakan pengguna `John` untuk mengambil peran `JohnRole1`. Untuk permintaan ini, dia menggunakan kredensial dari peran tersebut untuk mengambil peran `JohnRole2`. Ini dikenal sebagai perangkaian peran. Identitas sumber yang ia tetapkan ketika ia diasumsikan peran `John1` tetap ada dalam permintaan untuk mengasumsikan `JohnRole2`. Jika John mencoba untuk menetapkan identitas sumber yang berbeda ketika mengambil peran, permintaan ditolak. John memberikan dua tanda sesi ke dalam permintaan. Dia menetapkan kedua tanda itu sebagai transitif. Permintaan mewarisi tanda `Department` sebagai transitif karena John mengaturnya sebagai transitif ketika ia mengasumsikan `JohnRole1`. Untuk informasi lebih lanjut tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan. Untuk informasi lebih lanjut tentang kunci transitif dalam rantai peran, lihat Merangkai peran dengan tag sesi.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "John"
            },
            "sourceIdentity": "John"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
```

```
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
 Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "John",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
```

```
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
        "sourceIdentity": "John",
        "sessionTokenSize": 1652,
        "sessionTokenUtilization": 40
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

Contoh AWS layanan AWS STS Peristiwa API dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat oleh AWS layanan yang memanggil API layanan lain menggunakan izin dari peran layanan. Ini menunjukkan entri CloudTrail log untuk permintaan yang dibuat di akun 777788889999.

```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
```

```
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "amzn-s3-demo-bucket"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

Contoh SAML AWS STS Peristiwa API dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk AWS STS AssumeRoleWithSAML tindakan tersebut. Permintaan tersebut mencakup atribut SAML `CostCenter` dan `Project` yang diberikan melalui pernyataan SAML sebagai tanda sesi. Tanda itu diatur sebagai transitif sehingga mereka bertahan dalam skenario perangkaian peran. Permintaan menyertakan parameter API opsional`DurationSeconds`, direpresent `durationSeconds` asikan seperti dalam CloudTrail log, dan diatur ke `1800` detik. Permintaan juga menyertakan atribut SAML`sourceIdentity`, yang diteruskan dalam pernyataan SAML. Jika seseorang menggunakan kredensial sesi peran yang dihasilkan untuk mengambil peran lain, identitas sumber ini tetap ada.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
```

```
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 34,
        "sessionTokenSize": 1428,
        "sessionTokenUtilization": 34,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
```

```
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```


Contoh OIDC AWS STS Peristiwa API dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan yang dibuat untuk AWS STS AssumeRoleWithWebIdentity tindakan tersebut. Permintaan menyertakan atribut `CostCenter` dan `Project` yang dilewatkan melalui token penyedia identitas (IdP) OpenID Connect (OIDC) sebagai tag sesi. Tanda itu diatur sebagai transitif sehingga mereka bertahan dalam perangkaian peran. Permintaan tersebut mencakup atribut `sourceIdentity` dari token penyedia identitas. Jika seseorang menggunakan kredensial sesi peran yang dihasilkan untuk mengambil peran lain, identitas sumber ini tetap ada.

En CloudTrail tri log juga berisi `additionalEventData` bidang dengan `identityProviderConnectionVerificationMethod` atribut. Atribut ini menunjukkan metode yang AWS digunakan untuk memverifikasi koneksi dengan penyedia OIDC. Nilai atribut akan menjadi salah satu `IAMTrustStore` atau`Thumbprint`. `IAMTrustStore`Nilai menunjukkan bahwa AWS berhasil memverifikasi koneksi dengan IDP OIDC menggunakan pustaka otoritas sertifikat root

tepercaya (CA) kami. Thumb`print`Nilai menunjukkan bahwa AWS menggunakan sidik jari sertifikat yang disetel dalam konfigurasi IdP untuk memverifikasi sertifikat server IDC IdP.


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>:<id of application>:<id of user>",
    "userName": "<id of user>",
    "identityProvider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>"
  },
  "eventTime": "2024-07-09T15:41:37Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/2.13.29 Python/3.11.6 Windows/10 exe/AMD64 prompt/off command/
sts.assume-role-with-web-identity",
  "requestParameters": {
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "<assigned role session name>",
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "principalTags": {
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ]
  },
  "responseElements": {
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "sessionToken": "<encoded session token blob>",
      "expiration": "Jul 9, 2024, 4:41:37 PM"
    },
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "assumedRoleUser": {
```

```
      "assumedRoleId": "AROA123456789EXAMPLE:<assigned role session name>",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/<assigned
 role session name>"
    },
    "provider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>",
    "audience": "<id of application>",
    "sessionTokenSize": 1536,
    "sessionTokenUtilization": 37
  },
  "additionalEventData": {
    "identityProviderConnectionVerificationMethod": "IAMTrustStore"
  },
  "requestID": "aEXAMPLE-0b26-40df-8973-c7012EXAMPLE",
  "eventID": "aEXAMPLE-ee29-4ac0-a0ed-3f5c5EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "444455556666",
      "type": "AWS::IAM::Role",
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "444455556666",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
  }
}
```

Contoh AWS STS Acara API menggunakan titik akhir global dalam file CloudTrail log

Untuk permintaan ke AWS Security Token Service (AWS STS) titik akhir global (`https://sts.amazonaws.com`), ser AWS STS takan bidang AWS CloudTrail log tambahan: `endpointType` dan`awsServingRegion`. Bidang-bidang ini muncul di bawah `additionalEventData` `RequestDetails` elemen untuk mencatat jenis penyajian Wilayah AWS dan titik akhir yang dipanggil. Bid `endpointType` ang dapat memiliki nilai `global` atau `regional` untuk menunjukkan

jenis titik akhir global yang melayani permintaan. Untuk informasi selengkapnya tentang perubahan titik akhir AWS STS global, lihatAWS STS Wilayah dan titik akhir.

> **Note**
>
> AWS CloudTrail log untuk permintaan yang dibuat ke titik akhir AWS STS global akan dikirim ke Wilayah Timur AS (Virginia Utara). CloudTrail log untuk permintaan yang dilayani oleh titik akhir AWS STS Regional akan terus dicatat ke Wilayah masing-masing di CloudTrail.

Contoh berikut menunjukkan entri CloudTrail log untuk AWS STS permintaan yang dibuat ke titik akhir global (`https://sts.amazonaws.com`) yang berasal dari Wilayah Eropa (Stockholm) - eu-nord-1. Nilai `endpointType` bidang `global` menunjukkan bahwa AWS STS permintaan dilayani oleh titik akhir global di Wilayah Eropa (Stockholm).


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:48Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
```

```
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:48 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "awsServingRegion": "eu-north-1",
            "endpointType": "global"
        }
    },
    "requestID": "EXAMPLE7-2497-457a-9586-f21feEXAMPLE",
    "eventID": "EXAMPLEc-3d26-4c3a-9c94-722a9EXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
```

```
        "clientProvidedHostHeader": "sts-global.eu-north-1.amazonaws.com"
    }
}
```

Sebagai perbandingan, contoh berikut menunjukkan entri CloudTrail log untuk AWS STS permintaan yang dibuat ke titik akhir Regional (`https://sts.us-west-2.amazonaws.com`) yang dilayani oleh titik akhir Regional di Wilayah Eropa (Stockholm) - eu-utara-1. Nilai `endpointType` bidang `regional` menunjukkan bahwa AWS STS permintaan dilayani oleh titik akhir global di Wilayah Eropa (Stockholm).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:30Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "eu-north-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
```

```
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:30 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "endpointType": "regional",
            "awsServingRegion": "eu-north-1"
        }
    },
    "requestID": "EXAMPLEd-2116-4cd7-bd72-9f72fEXAMPLE",
    "eventID": "EXAMPLEd-219a-48ed-bc54-00e3cEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts.eu-north-1.amazonaws.com"
    }
}
```

Contoh peristiwa masuk di log CloudTrail

CloudTrail file log berisi peristiwa yang diformat menggunakan JSON. Peristiwa masuk mewakili permintaan masuk tunggal dan mencakup informasi tentang penanggung jawab masuk, Wilayah, serta tanggal dan waktu tindakan.

Contoh acara sukses masuk dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk acara masuk yang berhasil.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/John",
    "accountId": "111122223333",
    "userName": "John"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```


Contoh berikut menunjukkan entri CloudTrail log untuk permintaan kode otorisasi yang berhasil.


```
{
"eventVersion": "1.11",
```

```
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:thesjain-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/thesjain-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-17T22:50:14Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-17T22:51:32Z",
"eventSource": "signin.amazonaws.com",
"eventName": "AuthorizeOAuth2Access",
"awsRegion": "us-east-1",
"sourceIPAddress": "52.94.133.136",
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36",
"requestParameters": {
"scope": "openid",
"redirect_uri": "http://127.0.0.1:53037/oauth/callback",
"code_challenge_method": "SHA-256",
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "e2854c76-1cba-4360-9fd1-5037b591466b",
"eventID": "59e1720d-3deb-44ff-933d-6828be2a860a",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
```

```
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```

Contoh berikut menunjukkan entri CloudTrail log untuk permintaan pembuatan token OAuth2 yang berhasil.

```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:jacobjoj-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/jacobjoj-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-18T20:38:10Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-18T20:38:44Z",
"eventSource": "signin.amazonaws.com",
"eventName": "CreateOAuth2Token",
"awsRegion": "us-east-1",
"sourceIPAddress": "15.248.6.6",
"userAgent": "aws-cli/2.32.0 md/awscrt#0.28.4 ua/2.1 os/macos#24.6.0 md/arch#arm64
 lang/python#3.13.9 md/pyimpl#CPython m/b,AA,Z,E cfg/retry-mode#standard md/
installer#exe sid/35033f4ca1bd md/prompt#off md/command#login",
"requestParameters": {
```

```
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "94562943-c85b-4dc1-bf72-43b0fd42d6de",
"eventID": "0b338fac-6a10-4740-b34d-1bb6923e799e",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```


Untuk detail selengkapnya tentang informasi yang terkandung dalam file CloudTrail log, lihat CloudTrail Referensi Peristiwa di Panduan AWS CloudTrail Pengguna.

Contoh peristiwa kegagalan masuk dalam file CloudTrail log

Contoh berikut menunjukkan entri CloudTrail log untuk peristiwa masuk yang gagal.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
```

```
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

Dari informasi ini, Anda dapat menentukan bahwa upaya masuk dilakukan oleh pengguna IAM bernama `JaneDoe`, seperti yang ditunjukkan pada elemen `userIdentity`. Anda juga dapat melihat bahwa upaya masuk gagal, seperti yang ditunjukkan pada elemen `responseElements`. Anda dapat melihatnya bahwa `JaneDoe` mencoba masuk ke konsol Amazon SNS pada pukul 17.35 (UTC) pada 8 Juli 2014.

Contoh peristiwa gagal masuk yang disebabkan oleh nama pengguna yang salah

Contoh berikut menunjukkan entri CloudTrail log untuk peristiwa masuk yang gagal yang disebabkan oleh pengguna memasukkan nama pengguna yang salah. AWS menutupi `userName` teks `HIDDEN_DUE_TO_SECURITY_REASONS` untuk membantu mencegah pengekspos informasi yang berpotensi sensitif.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
```

```
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```

Perilaku kebijakan kepercayaan peran IAM

Pada tanggal 21 September 2022, AWS membuat perubahan pada perilaku kebijakan kepercayaan peran IAM untuk mewajibkan izin eksplisit dalam kebijakan kepercayaan peran ketika peran mengambil dirinya sendiri. Peran IAM dalam daftar izin perilaku lama memiliki EventData bidang tambahan yang ada untuk eksplisit TrustGrant untuk `AssumeRole` acara. Nilai false ketika peran pada daftar izinkan warisan mengasumsikan dirinya menggunakan perilaku warisan. `explicitTrustGrant` Ketika peran pada daftar izin lama mengasumsikan dirinya sendiri tetapi perilaku kebijakan kepercayaan peran telah diperbarui untuk secara eksplisit memungkinkan peran untuk mengambil dirinya sendiri, nil `explicitTrustGrant` ainya benar.

Hanya sejumlah kecil peran IAM yang ada dalam daftar izin untuk perilaku lama, dan bidang ini hanya ada di CloudTrail log untuk peran ini ketika mereka mengambil diri mereka sendiri. Dalam kebanyakan kasus, peran IAM tidak perlu mengambil dirinya sendiri. AWS merekomendasikan memperbarui proses, kode, atau konfigurasi Anda untuk menghapus perilaku ini atau memperbarui kebijakan kepercayaan peran Anda untuk secara eksplisit mengizinkan perilaku ini. Untuk informasi selengkapnya, lihat Mengumumkan pembaruan perilaku kebijakan kepercayaan peran IAM.

Lacak tugas istimewa di AWS CloudTrail

Akun AWS Organizations manajemen atau akun administrator yang didelegasikan untuk IAM dapat melakukan beberapa tugas pengguna root pada akun anggota menggunakan akses root jangka pendek. Sesi istimewa jangka pendek memberi Anda kredensi sementara yang dapat Anda

lingkup untuk mengambil tindakan istimewa pada akun anggota di organisasi Anda. Anda dapat menggunakan langkah-langkah berikut untuk mengidentifikasi tindakan yang diambil oleh akun manajemen atau administrator yang didelegasikan selama `sts:AssumeRoot`sesi berlangsung.

> **Note**
>
> Titik akhir global tidak didukung untuk`sts:AssumeRoot`. CloudTrail mencatat `ConsoleLogin` peristiwa di Wilayah yang ditentukan untuk titik akhir.

Untuk melacak tindakan yang dilakukan oleh sesi istimewa di log CloudTrail

1. Temukan `AssumeRoot` acara di CloudTrail log Anda. Acara ini dihasilkan ketika akun manajemen Anda atau administrator yang didelegasikan untuk IAM mendapatkan satu set kredensi jangka pendek dari. `sts:AssumeRoot`

   Dalam contoh berikut, CloudTrail acara untuk AssumeRoot dicatat di `eventName` bidang.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/John",
                "accountId": "111111111111",
                "userName": "Admin2"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-10-25T20:45:28Z",
                "mfaAuthenticated": "false"
            },
            "assumedRoot": "true"
        }
    },
```

```
    "eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
        }
    }
}
```

Untuk langkah-langkah untuk mengakses CloudTrail log Anda, lihat Mendapatkan dan melihat file CloudTrail log Anda di Panduan AWS CloudTrail Pengguna.

2. Dalam log CloudTrail peristiwa, cari `targetPrincipal` yang menentukan tindakan akun anggota yang diambil, dan `accessKeyId` yang unik untuk `AssumeRoot` sesi.

   Dalam contoh berikut, `targetPrincipal` adalah 22222222222222 dan adalah CONTOH. `accessKeyId` ASIAIOSFODNN7

```
"eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
```

```
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
    }
}
```

3. Dalam CloudTrail log untuk prinsipal target, cari ID kunci akses yang sesuai dengan `accessKeyId` nilai dari `AssumeRoot` acara tersebut. Gunakan nilai `eventName` bidang untuk menentukan tugas istimewa yang dilakukan selama `AssumeRoot` sesi. Mungkin ada beberapa tugas istimewa yang dilakukan dalam satu sesi. Durasi sesi maksimum `AssumeRoot` adalah 900 detik (15 menit).

   Dalam contoh berikut, akun manajemen atau administrator yang didelegasikan menghapus kebijakan berbasis sumber daya untuk bucket Amazon S3.

```
{
    "eventVersion": "1.10",
    "userIdentity": {
        "type": "Root",
        "principalId": "222222222222",
        "arn": "arn:aws:iam::222222222222:root",
        "accountId": "222222222222",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2024-10-25T20:52:11Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-10-25T20:53:47Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "DeleteBucketPolicy",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "bucketName": "resource-policy-John",
        "Host": "resource-policy-John.s3.amazonaws.com",
        "policy": ""
    },
    "responseElements": null,
    "requestID": "1234567890abcdef0",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "readOnly": false,
```

```
    "resources": [
        {
            "accountId": "222222222222",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::resource-policy-John"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "222222222222",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "resource-policy-John.s3.amazonaws.com"
    }
}
```


Validasi kepatuhan untuk AWS Identity and Access Management

Third-party auditor menilai keamanan dan kepatuhan AWS Identity and Access Management (IAM) sebagai bagian dari beberapa program AWS kepatuhan. Ini mencakup SOC, PCI, FedRAMP, ISO, dan lainnya.

Pengalaman AWS pendaftaran baru kami tidak dirancang untuk beban kerja yang diatur. Jika Anda menggunakan pengalaman AWS pendaftaran baru kami, tetapi ingin menggunakannya AWS untuk beban kerja yang diatur, Anda dapat mendaftar untuk AWS (lanjutan) atau mengaktifkan fitur lanjutan untuk lingkungan Anda AWS .

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup berdasarkan Program Kepatuhan dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program Kep AWS atuhan Program Jaminan .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. Untuk informasi selengkapnya tentang tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS, lihat Dokumentasi AWS Keamanan.

Ketahanan di AWS Identity and Access Management

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah memiliki beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

AWS Identity and Access Management (IAM) dan AWS Security Token Service (AWS STS) adalah layanan mandiri berbasis Wilayah yang tersedia secara global.

IAM adalah hal yang kritis Layanan AWS. Setiap operasi yang dilakukan AWS harus diautentikasi dan disahkan oleh IAM. IAM memeriksa setiap permintaan terhadap identitas dan kebijakan yang disimpan di IAM untuk menentukan apakah permintaan tersebut diizinkan atau ditolak. IAM dirancang dengan bidang kontrol dan bidang data terpisah sehingga layanan mengautentikasi bahkan selama kegagalan yang tidak terduga. Sumber daya IAM yang digunakan dalam otorisasi, seperti peran dan kebijakan, disimpan di bidang kontrol. Pelanggan IAM dapat mengubah konfigurasi sumber daya ini dengan menggunakan operasi IAM seperti `DeletePolicy` dan. `AttachRolePolicy` Permintaan perubahan konfigurasi tersebut masuk ke bidang kontrol. Ada satu pesawat kontrol IAM untuk semua komersial Wilayah AWS, yang terletak di Wilayah AS Timur (Virginia N.). Sistem IAM kemudian menyebarkan perubahan konfigurasi ke bidang data IAM di setiap diaktifkan. Wilayah AWS Bidang data IAM pada dasarnya adalah replika hanya-baca dari data konfigurasi bidang kontrol IAM. Masing-masing Wilayah AWS memiliki instance yang sepenuhnya independen dari pesawat data IAM, yang melakukan otentikasi dan otorisasi untuk permintaan dari Wilayah yang sama. Di setiap Wilayah, pesawat data IAM didistribusikan di setidaknya tiga Availability Zone, dan memiliki kapasitas yang cukup untuk mentolerir hilangnya Availability Zone tanpa gangguan pelanggan. Baik kontrol IAM dan pesawat data dibangun untuk nol waktu henti yang direncanakan, dengan semua pembaruan perangkat lunak dan operasi penskalaan dilakukan dengan cara yang tidak terlihat oleh pelanggan.

Di Wilayah yang diaktifkan secara default, permintaan ke titik akhir AWS STS global secara otomatis ditayangkan di Wilayah yang sama tempat permintaan berasal. Di Wilayah keikutsertaan, permintaan ke titik akhir AWS STS global dilayani oleh satu Wilayah AWS, US East (Virginia N.). Anda dapat menggunakan AWS STS endpoint Regional untuk mengurangi latensi atau memberikan redundansi tambahan untuk aplikasi Anda. Untuk mempelajari selengkapnya, lihat Kelola AWS STS dalam sebuah Wilayah AWS.

Peristiwa tertentu dapat mengganggu komunikasi antara Wilayah AWS melalui jaringan. Namun, bahkan ketika Anda tidak dapat berkomunikasi dengan endpoint IAM global, masih AWS STS dapat mengautentikasi prinsip-prinsip IAM dan IAM dapat mengotorisasi permintaan Anda. Detail spesifik

dari suatu peristiwa yang mengganggu komunikasi akan menentukan kemampuan Anda untuk mengakses AWS layanan. Dalam kebanyakan situasi, Anda dapat terus menggunakan kredensil IAM di lingkungan Anda. AWS Kondisi berikut mungkin berlaku untuk peristiwa yang mengganggu komunikasi.

Kunci akses untuk pengguna IAM

Anda dapat mengautentikasi tanpa batas waktu di Wilayah dengan kunci akses jangka panjang untuk pengguna IAM. Saat Anda menggunakan AWS Command Line Interface dan APIs, Anda dapat memberikan kunci AWS akses sehingga AWS dapat memverifikasi identitas Anda dalam permintaan terprogram.

> **Important**
>
> Sebagai praktik terbaik, kami menyarankan agar pengguna Anda masuk dengan kredensi sementara, bukan kunci akses jangka panjang.

Kredensial sementara

Anda dapat meminta kredensi sementara baru dengan titik akhir layanan AWS STS Regional setidaknya selama 24 jam. Operasi API berikut menghasilkan kredensil sementara.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAM
- GetFederationToken
- GetSessionToken

Prinsipal dan izin

- Anda mungkin tidak dapat menambahkan, memodifikasi, atau menghapus prinsip atau izin di IAM.
- Kredensi Anda mungkin tidak mencerminkan perubahan pada izin yang baru saja Anda terapkan di IAM. Untuk informasi selengkapnya, lihat Perubahan yang saya buat tidak selalu langsung terlihat.

Konsol Manajemen AWS

- Anda mungkin dapat menggunakan titik akhir masuk Regional untuk masuk ke pengguna IAM Konsol Manajemen AWS sebagai. Titik akhir masuk regional memiliki format URL berikut.

```
https://{Account ID}.signin.aws.amazon.com/console?region={Region}
```

Contoh: https://111122223333.signin.aws.amazon.com /console? wilayah=us-barat-2

- Anda mungkin tidak dapat menyelesaikan otentikasi multi-faktor Universal 2nd Factor (U2F) (MFA).

Praktik terbaik untuk ketahanan IAM

AWS telah membangun ketahanan ke dalam Wilayah AWS dan Availability Zones. Ketika Anda mengamati praktik terbaik IAM berikut dalam sistem yang berinteraksi dengan lingkungan Anda, Anda memanfaatkan ketahanan itu.

1. Gunakan endpoint layanan AWS STS Regional, bukan endpoint global default.
2. Tinjau konfigurasi lingkungan Anda untuk sumber daya vital yang secara rutin membuat atau memodifikasi sumber daya IAM, dan menyiapkan solusi fallback yang menggunakan sumber daya IAM yang ada.

Keamanan infrastruktur di AWS Identity and Access Management

Sebagai layanan terkelola, AWS Identity and Access Management dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses IAM melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

IAM dapat diakses secara terprogram dengan menggunakan API HTTPS IAM, yang memungkinkan Anda mengeluarkan permintaan HTTPS secara langsung ke layanan. Query API mengembalikan informasi sensitif, termasuk kredensial keamanan. Oleh karena itu, Anda harus menggunakan

HTTPS untuk semua permintaan API. Saat Anda menggunakan API HTTPS, Anda harus menyertakan kode untuk menandatangani permintaan secara digital menggunakan kredensial Anda.

Anda dapat menghubungi operasi API ini dari lokasi jaringan mana pun, tetapi IAM mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan IAM untuk mengontrol akses dari titik akhir Amazon Virtual Private Cloud (Amazon VPC) tertentu atau spesifik. VPCs Secara efektif, ini mengisolasi akses jaringan ke sumber daya IAM tertentu hanya dari VPC tertentu dalam jaringan. AWS

Konfigurasi dan analisis kerentanan di AWS Identity and Access Management

AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patching database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut:

- Model Tanggung Jawab Bersama
- Amazon Web Services: Gambaran Umum Proses Keamanan (whitepaper)

Sumber daya berikut juga membahas konfigurasi dan analisis kerentanan di AWS Identity and Access Management (IAM):

- Validasi kepatuhan untuk AWS Identity and Access Management
- Praktik terbaik keamanan dan kasus penggunaan di AWS Identity and Access Management

AWS kebijakan yang dikelola untuk AWS Identity and Access Management, Penganalisis Akses IAM, dan manajer akses akun

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS kebijakan terkelola dirancang untuk memberikan izin untuk banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan ter AWS kelola mungkin tidak memberikan izin hak istimewa terkecil untuk kasus penggunaan spesifik Anda karena kebijakan tersebut tersedia untuk digunakan semua

AWS pelanggan. Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan ter AWS kelola ketika yang baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

IAMReadOnlyAccess

Gunakan kebijakan ter `IAMReadOnlyAccess` kelola untuk mengizinkan akses baca saja ke sumber daya IAM. Kebijakan ini memberikan izin untuk mendapatkan dan mencantumkan semua sumber daya IAM. Ini memungkinkan melihat detail dan laporan aktivitas untuk pengguna, grup, peran, kebijakan, penyedia identitas, dan perangkat MFA. Ini tidak termasuk kemampuan untuk membuat atau menghapus sumber daya atau akses ke sumber daya Penganalisis Akses IAM. Lihat kebijakan untuk daftar lengkap layanan dan tindakan yang didukung oleh kebijakan ini.

IAMUserChangePassword

Gunakan kebijakan ter `IAMUserChangePassword` kelola untuk mengizinkan pengguna IAM mengubah kata sandi mereka.

Anda mengonfigurasi setelan Akun IAM dan kebijakan Kata Sandi untuk mengizinkan pengguna IAM mengubah kata sandi akun IAM mereka. Saat Anda mengizinkan tindakan ini, IAM melampirkan kebijakan ke setiap pengguna.

Untuk melihat kebijakan JSON, lihat IAMUserChangePassword di Panduan Referensi AWS Kebijakan Terkelola.

IAMAccessAnalyzerFullAccess

Gunakan kebijakan ter `IAMAccessAnalyzerFullAccess` AWS kelola untuk mengizinkan administrator mengakses Penganalisis Akses IAM.

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- IAM Access Analyzer - Memungkinkan izin administratif penuh untuk semua sumber daya di IAM Access Analyzer.
- Buat peran terkait layanan — Memungkinkan administrator untuk membuat peran terkait layanan, yang memungkinkan Penganalisis Akses IAM menganalisis sumber daya di layanan lain atas nama Anda. Izin ini memungkinkan pembuatan peran terkait layanan hanya untuk digunakan oleh IAM Access Analyzer.
- AWS Organizations- Memungkinkan administrator untuk menggunakan Penganalisis Akses IAM untuk organisasi di AWS Organizations. Setelah mengaktifkan akses tepercaya untuk IAM Access Analyzer di AWS Organizations, anggota akun manajemen dapat melihat temuan di seluruh organisasi mereka.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
```

```
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

Gunakan kebijakan ter `IAMAccessAnalyzerReadOnlyAccess` AWS kelola untuk mengizinkan akses read-only ke IAM Access Analyzer.

Untuk juga mengizinkan akses read-only ke IAM Access Analyzer untuk AWS Organizations, buat kebijakan yang dikelola pelanggan yang mengizinkan tindakan Deskripsikan dan Daftar dari kebijakan terkel IAMAccessAnalyzerFullAccess AWS ola.

Service-level izin

Kebijakan ini menyediakan akses read-only ke IAM Access Analyzer. Tidak ada izin layanan lain yang disertakan dalam kebijakan ini.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
```

```
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

Anda tidak dapat melampirkan AccessAnalyzerServiceRolePolicy ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran terkait layanan yang memungkinkan IAM Access Analyzer melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk AWS Identity and Access Management dan Penganalisis Akses.

Pengelompokan izin

Kebijakan ini memungkinkan akses ke IAM Access Analyzer untuk menganalisis metadata sumber daya dari beberapa Layanan AWS sumber daya.

- Amazon DynamoDB — Memungkinkan izin untuk melihat aliran dan tabel DynamoDB.
- Amazon Elastic Compute Cloud — Memungkinkan izin untuk menggambarkan alamat IP, snapshot, dan VPC.
- Amazon Elastic Container Registry - Memungkinkan izin untuk menjelaskan repositori gambar, mengambil pengaturan akun, dan mengambil kebijakan registri dan repositori.
- Amazon Elastic File System — Memungkinkan izin untuk melihat deskripsi sistem file Amazon EFS dan melihat kebijakan tingkat sumber daya untuk sistem file Amazon EFS.
- AWS Identity and Access Management- Memungkinkan izin untuk mengambil informasi tentang peran tertentu dan daftar peran IAM yang memiliki awalan jalur tertentu. Memungkinkan izin untuk mengambil informasi tentang pengguna, grup IAM, profil login, kunci akses, dan data layanan yang terakhir diakses.
- AWS Key Management Service- Mengizinkan izin untuk melihat informasi terperinci tentang kunci KMS dan kebijakan dan hibah utamanya.
- AWS Lambda- Mengizinkan izin untuk melihat informasi tentang alias, fungsi, lapisan, dan alias Lambda.

- AWS Organizations- Mengizinkan izin ke AWS Organizations dan memungkinkan pembuatan penganalisis dalam AWS organisasi sebagai zona kepercayaan.
- Layanan Database Relasional Amazon — Memungkinkan izin untuk melihat informasi terperinci tentang snapshot Amazon RDS DB dan snapshot cluster Amazon RDS DB.
- Amazon Simple Storage Service — Memungkinkan izin untuk melihat informasi terperinci tentang titik akses Amazon S3, bucket, titik akses bucket direktori Amazon S3, dan bucket direktori.
- AWS Secrets Manager— Memungkinkan izin untuk melihat informasi terperinci tentang rahasia dan kebijakan sumber daya yang dilampirkan pada rahasia.
- Layanan Pemberitahuan Sederhana Amazon - Memungkinkan izin untuk melihat informasi terperinci tentang suatu topik.
- Amazon Simple Queue Service — Memungkinkan izin untuk melihat informasi terperinci tentang antrian tertentu.

Untuk melihat kebijakan JSON, lihat AccessAnalyzerServiceRolePolicy di Panduan Referensi AWS Kebijakan Terkelola.

IAMAuditRootUserCredentials

Gunakan kebijakan `IAMAuditRootUserCredentials` AWS terkelola untuk mengurangi izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota untuk mengaudit status kredensi pengguna root dari akun anggota. Anda dapat membuat daftar atau mendapatkan informasi kredensi pengguna root individu seperti:

- Apakah ada kata sandi pengguna root
- Jika pengguna root memiliki kunci akses dan kapan terakhir digunakan
- Jika pengguna root memiliki sertifikat penandatanganan terkait
- Perangkat MFA terkait pengguna root
- Daftar status kredensi pengguna root yang dikonsolidasikan

Anda tidak dapat melampirkan `IAMAuditRootUserCredentials` ke entitas IAM Anda. Kebijakan ini dilampirkan AssumeRoot untuk melakukan tugas istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices",
            "iam:GetAccountSummary",
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyAuditingCredentialsOnNonRootUserResource",
         "Action": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices" ,
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- DenyAllOtherActionsOnAnyResource— Menolak akses ke kredenSIAL untuk semua sumber daya.

- DenyAuditingCredentialsOnNonRootUserResource— Menolak akses ke kredensional untuk semua sumber daya pengguna non-root.

IAMCreateRootUserPassword

Gunakan kebijakan `IAMCreateRootUserPassword` AWS terkelola untuk mengurangi izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota untuk mengizinkan pemulihan kata sandi untuk akun anggota tanpa kredentif pengguna root.

Anda tidak dapat melampirkan `IAMCreateRootUserPassword` ke entitas IAM Anda. Kebijakan ini dilampirkan AssumeRoot untuk melakukan tugas istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyCreatingPasswordOnNonRootUserResource",
         "Action": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- DenyAllOtherActionsOnAnyResource— Menolak akses untuk mendapatkan atau membuat kata sandi untuk semua sumber daya.
- DenyCreatingPasswordOnNonRootUserResource— Menolak akses untuk mendapatkan atau membuat kata sandi untuk semua sumber daya pengguna non-root.

IAMDeleteRootUserCredentials

Gunakan kebijakan `IAMDeleteRootUserCredentials` AWS terkelola untuk mengurangi izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota untuk menghapus kredentif pengguna root termasuk kata sandi, kunci akses, sertifikat penandatanganan, dan menonaktifkan MFA. Izin tambahan diperlukan untuk tindakan istimewa ini, sehingga Anda dapat melihat informasi kredensi yang terakhir digunakan, memverifikasi informasi yang terakhir digunakan untuk pengguna root akun anggota, dan membuat daftar izin untuk semua kredensional pengguna root yang akan dihapus.

Anda tidak dapat melampirkan `IAMDeleteRootUserCredentials` ke entitas IAM Anda. Kebijakan ini dilampirkan AssumeRoot untuk melakukan tugas istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "DenyAllOtherActionsOnAnyResource",
   "Effect": "Deny",
   "NotAction": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
```

```
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "Resource": "*"
  },
  {
   "Sid": "DenyDeletingRootUserCredentialsOnNonRootUserResource",
   "Effect": "Deny",
   "Action": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "NotResource": "arn:aws:iam::*:root"
  }
 ]
}
```

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- DenyAllOtherActionsOnAnyResource— Menolak akses untuk mendapatkan atau menghapus kredenSIAL untuk semua sumber daya.
- DenyDeletingRootUserCredentialsOnNonRootUserResource— Menolak akses untuk mendapatkan atau menghapus kredenSIAL untuk semua sumber daya pengguna non-root.

S3UnlockBucketPolicy

Gunakan kebijakan `S3UnlockBucketPolicy` AWS terkelola untuk mengurangi izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota untuk menghapus kebijakan bucket yang salah dikonfigurasi yang menolak semua prinsipal mengakses bucket Amazon S3.

Anda tidak dapat melampirkan `S3UnlockBucketPolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan AssumeRoot untuk melakukan tugas istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyManagingBucketPolicyForNonRootCallers",
         "Action": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*",
         "Condition" : {
            "ArnNotLike" : {
               "aws:PrincipalArn" : "arn:aws:iam::*:root"
            }
         }
      }
```

```
    ]
}
```

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- DenyAllOtherActionsOnAnyResource— Menolak akses ke kebijakan bucket untuk semua sumber daya.
- DenyManagingBucketPolicyForNonRootCallers— Menolak akses ke kebijakan bucket untuk semua sumber daya pengguna non-root.

SQSUnlockQueuePolicy

Gunakan kebijakan `SQSUnlockQueuePolicy` AWS terkelola untuk mengurangi izin saat Anda melakukan tugas istimewa pada akun AWS Organizations anggota untuk menghapus kebijakan berbasis sumber daya Amazon Simple Queue Service yang menolak semua kepala sekolah mengakses antrian Amazon SQS.

Anda tidak dapat melampirkan `SQSUnlockQueuePolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan AssumeRoot untuk melakukan tugas istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

Pengelompokan izin

Kebijakan ini dikelompokkan ke dalam pernyataan berdasarkan kumpulan izin yang diberikan.

- DenyAllOtherActionsOnAnyResource— Menolak akses ke tindakan Amazon SQS untuk semua sumber daya.
- DenyGettingQueueAttributesOnNonOwnQueue— Menolak akses ke atribut antrian Amazon SQS untuk antrian yang dimiliki oleh akun lain.
- DenyActionsForNonRootUser— Menolak akses ke tindakan Amazon SQS untuk semua sumber daya pengguna non-root.

AccountAccessManagerServiceRolePolicy

Anda tidak dapat melampirkan `AccountAccessManagerServiceRolePolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran terkait layanan yang memungkinkan pengelola akses akun melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat the section called "Keamanan".

Service-level izin

Kebijakan ini memungkinkan pengelola akses akun untuk mengambil informasi tentang akun Organisasi Anda untuk mempertahankan keanggotaan akun yang akurat untuk manajemen akses. Manajer akses akun menggunakan izin ini untuk mendeteksi kapan akun meninggalkan organisasi Anda, memverifikasi apakah akses layanan tepercaya diaktifkan, dan mengonfirmasi status akun sebelum mengelola hak.

- AWS Organizations- Mengizinkan izin untuk membuat daftar akun di organisasi, memverifikasi apakah akses layanan tepercaya diaktifkan, dan menjelaskan status akun.

Untuk melihat kebijakan JSON, lihat AccountAccessManagerServiceRolePolicy di Panduan Referensi AWS Kebijakan Terkelola.

IAM, IAM Access Analyzer, dan pembaruan manajer akses akun ke AWS kebijakan terkelola

Lihat detail tentang pembaruan IAM dan kebijakan ter AWS kelola sejak layanan mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen Penganalisis Akses IAM dan IAM.

Ubah	Deskripsi	Date
AccountAccessManagerServiceRolePolicy — Ditambahkan kebijakan terkelola	Manajer akses akun menambahkan kebijakan peran terkait layanan yang memungkinkan layanan mengambil informasi tentang akun Organisasi Anda.	Juli 31, 2026

Ubah	Deskripsi	Date
IAMUserChangePassword— Menambahkan izin	IAM menambahkan izin `IAMUserChangePassword` untuk mengizinkan pengguna yang ditentukan dalam jalur.	Mei 28, 2025
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer ditambahkan `iam:GetAccountAuthorizationDetails` ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	12 Mei 2025
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk titik akses bucket direktori Amazon S3 ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	Maret 31, 2025
IAMDeleteRootUserCredentials— Izin yang dihapus	IAM menghapus `iam:DeleteVirtualMFADevice` izin dari kebijakan terkelola.	Januari 7, 2025
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengaturan akun Amazon ECR dan kebijakan registri ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	Desember 10, 2024

Ubah	Deskripsi	Date
IAMAuditRootUserCredentials — Ditambahkan kebijakan terkelola	IAM menambahkan kebijakan terkelola untuk Mengelola akses root secara terpusat untuk akun anggota untuk mencakup tugas istimewa yang dapat Anda lakukan pada akun AWS Organizations anggota.	14 November 2024
IAMCreateRootUserPassword— Ditambahkan kebijakan terkelola	IAM menambahkan kebijakan terkelola untuk Mengelola akses root secara terpusat untuk akun anggota untuk mencakup tugas istimewa yang dapat Anda lakukan pada akun AWS Organizations anggota.	14 November 2024
IAMDeleteRootUserCredentials— Ditambahkan kebijakan terkelola	IAM menambahkan kebijakan terkelola untuk Mengelola akses root secara terpusat untuk akun anggota untuk mencakup tugas istimewa yang dapat Anda lakukan pada akun AWS Organizations anggota.	14 November 2024
S3UnlockBucketPolicy— Ditambahkan kebijakan terkelola	IAM menambahkan kebijakan terkelola untuk Mengelola akses root secara terpusat untuk akun anggota untuk mencakup tugas istimewa yang dapat Anda lakukan pada akun AWS Organizations anggota.	14 November 2024

Ubah	Deskripsi	Date
SQSUnlockQueuePolicy— Ditambahkan kebijakan terkelola	IAM menambahkan kebijakan terkelola untuk Mengelola akses root secara terpusat untuk akun anggota untuk mencakup tugas istimewa yang dapat Anda lakukan pada akun AWS Organizations anggota.	14 November 2024
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengguna IAM dan tag peran ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	Oktober 29, 2024
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengguna IAM dan kebijakan peran ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	30 Mei 2024

Ubah	Deskripsi	Date
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari akses publik blok untuk snapshot Amazon EC2 ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	23 Januari 2024
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk aliran dan tabel DynamoDB ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	Januari 11, 2024
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk bucket direktori Amazon S3 ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	1 Desember 2023

Ubah	Deskripsi	Date
IAMAccessAnalyzerReadOnlyAccess— Menambahkan izin	IAM Access Analyzer menambahkan izin untuk memungkinkan Anda memeriksa apakah pembaruan kebijakan Anda memberikan akses tambahan. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	26 November 2023
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan tindakan IAM ke izin tingkat layanan `AccessAnalyzerServiceRolePolicy` untuk mendukung tindakan berikut: • Mendaftarkan entitas untuk kebijakan • Menghasilkan detail layanan yang terakhir diakses • Informasi kunci akses daftar	26 November 2023

Ubah	Deskripsi	Date
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk jenis sumber daya berikut ke izin tingkat layanan: `AccessAnalyzerServiceRolePolicy` • Snapshot volume Amazon EBS • Repositori Amazon ECR • Sistem file Amazon EFS • Snapshot DB Amazon RDS • Snapshot cluster Amazon RDS DB • Topik Amazon SNS	25 Oktober 2022
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan `lambda:GetFunctionUrlConfig` tindakan ke izin tingkat layanan. `AccessAnalyzerServiceRolePolicy`	April 6, 2022
AccessAnalyzerServiceRolePolicy— Menambahkan izin	IAM Access Analyzer menambahkan tindakan Amazon S3 baru untuk menganalisis metadata yang terkait dengan titik akses multi-wilayah.	2 September 2021

Ubah	Deskripsi	Date
IAMAccessAnalyzerReadOnlyAccess—Menambahkan izin	IAM Access Analyzer menambahkan tindakan baru untuk memberikan `ValidatePolicy` izin agar Anda dapat menggunakan pemeriksaan kebijakan untuk validasi. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	16 Maret 2021
IAM Access Analyzer mulai melacak perubahan	IAM Access Analyzer mulai melacak perubahan untuk kebijakan yang AWS dikelolanya.	1 Maret 2021

Fitur keamanan di luar IAM

Anda menggunakan IAM untuk mengontrol akses ke tugas yang dilakukan menggunakan Konsol Manajemen AWS, Alat Baris AWS Perintah, atau operasi API layanan menggunakan. AWS SDKs Beberapa produk AWS juga memiliki cara lain untuk mengamankan sumber dayanya. Daftar berikut ini memberikan beberapa contoh, meskipun tidak lengkap.

Amazon EC2

Di Amazon Elastic Compute Cloud, Anda masuk ke suatu instans dengan key pair (untuk instans Linux) atau menggunakan nama pengguna dan kata sandi (untuk instans Microsoft Windows).

Untuk informasi selengkapnya, lihat dokumentasi berikut ini:

- Memulai Instans Amazon EC2 Linux di EC2 Panduan Pengguna Amazon
- Memulai Instans Amazon EC2 Windows di EC2 Panduan Pengguna Amazon

Amazon RDS

Di Amazon Relational Database Service, Anda masuk ke mesin basis data dengan nama pengguna dan kata sandi yang terikat dengan basis data tersebut.

Untuk informasi selengkapnya, lihat Memulai dengan Amazon RDS di Panduan Pengguna Amazon RDS.

Amazon EC2 dan Amazon RDS

Di Amazon EC2 dan Amazon RDS Anda menggunakan grup keamanan untuk mengontrol lalu lintas ke instans atau database.

Untuk informasi selengkapnya, lihat dokumentasi berikut ini:

- Grup EC2 Keamanan Amazon untuk Instans Linux di EC2 Panduan Pengguna Amazon
- Grup EC2 Keamanan Amazon untuk Instans Windows di EC2 Panduan Pengguna Amazon
- Grup Keamanan Amazon RDS di Panduan Pengguna Amazon RDS

WorkSpaces

Di Amazon WorkSpaces, pengguna masuk ke desktop dengan nama pengguna dan kata sandi.

Untuk informasi selengkapnya, lihat Memulai WorkSpaces di Panduan WorkSpaces Administrasi Amazon.

Metode kontrol akses ini bukan bagian dari IAM. IAM memungkinkan Anda mengontrol cara AWS produk ini dikelola — membuat atau mengakhiri EC2 instans Amazon, menyiapkan desktop baru WorkSpaces, dan sebagainya. Artinya, IAM membantu Anda mengontrol tugas yang dilakukan dengan membuat permintaan ke Amazon Web Services, dan ini membantu Anda mengontrol akses ke Konsol Manajemen AWS. Namun, IAM tidak membantu Anda mengelola keamanan untuk tugas-tugas seperti masuk ke sistem operasi (Amazon EC2), database (Amazon RDS), atau desktop (Amazon WorkSpaces).

Saat Anda bekerja dengan AWS produk tertentu, pastikan untuk membaca dokumentasi untuk mempelajari opsi keamanan untuk semua sumber daya milik produk tersebut.

Penggunaan AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menyediakan kemampuan berikut:

- Penganalisis akses eksternal IAM Access Analyzer membantu mengidentifikasi sumber daya di organisasi dan akun Anda yang dibagikan dengan entitas eksternal.
- Penganalisis akses internal IAM Access Analyzer membantu mengidentifikasi akses internal ke sumber daya yang dipilih di organisasi dan akun Anda.
- Penganalisis akses IAM Access Analyzer yang tidak digunakan membantu mengidentifikasi akses yang tidak digunakan di organisasi dan akun Anda.
- IAM Access Analyzer memvalidasi kebijakan IAM terhadap tata bahasa kebijakan dan AWS praktik terbaik.
- Pemeriksaan kebijakan kustom IAM Access Analyzer membantu memvalidasi kebijakan IAM terhadap standar keamanan yang Anda tentukan.
- IAM Access Analyzer menghasilkan kebijakan IAM berdasarkan aktivitas akses di log Anda AWS CloudTrail .

Mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal

IAM Access Analyzer membantu Anda mengidentifikasi sumber daya di organisasi dan akun Anda, seperti bucket Amazon S3 atau peran IAM, yang dibagikan dengan entitas eksternal. Hal ini memungkinkan Anda mengidentifikasi akses yang tidak diinginkan ke sumber daya dan data Anda, yang merupakan sebuah risiko keamanan. IAM Access Analyzer mengidentifikasi sumber daya yang dibagikan dengan prinsipal eksternal dengan menggunakan penalaran berbasis logika untuk menganalisis kebijakan berbasis sumber daya di lingkungan Anda. AWS Untuk setiap instance sumber daya yang dibagikan di luar akun Anda, IAM Access Analyzer menghasilkan temuan. Temuan mencakup informasi tentang akses dan prinsipal eksternal yang diberikan kepadanya. Anda dapat meninjau temuan untuk menentukan apakah akses dimaksudkan dan aman atau apakah akses tidak diinginkan dan berisiko keamanan. Selain membantu mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal, Anda dapat menggunakan temuan Penganalisis Akses IAM untuk melihat bagaimana kebijakan memengaruhi akses publik dan lintas akun ke sumber daya sebelum menerapkan izin sumber daya. Temuan diatur dalam dasbor ringkasan visual. Dasbor

menyoroti pemisahan antara temuan akses publik dan lintas akun, dan memberikan rincian temuan berdasarkan jenis sumber daya. Untuk mempelajari lebih lanjut tentang dasbor, lihatMelihat dasbor temuan Penganalisis Akses IAM.

> **Note**
>
> Entitas eksternal dapat berupa AWS akun lain, pengguna root, pengguna atau peran IAM, pengguna federasi, pengguna anonim, atau entitas lain yang dapat Anda gunakan untuk membuat filter. Untuk informasi lebih lanjut, lihat Elemen Kebijakan JSON AWS : Penanggung Jawab

Untuk petunjuk tentang cara memulai, lihatBuat penganalisis akses eksternal IAM Access Analyzer. Setelah Anda membuat penganalisis, Anda bisaKelola penganalisis akses eksternal IAM Access Analyzer.

Saat mengaktifkan Penganalisis Akses IAM, Anda membuat penganalisis untuk seluruh organisasi atau akun Anda. Akun atau organisasi yang Anda pilih dikenal sebagai zona kepercayaan untuk penganalisis. Penganalisis memantau semua sumber daya yang didukung dalam zona kepercayaan Anda. Setiap akses ke sumber daya oleh kepala sekolah dalam zona kepercayaan Anda dianggap tepercaya. Setelah diaktifkan, IAM Access Analyzer menganalisis kebijakan yang diterapkan ke semua sumber daya yang didukung di zona kepercayaan Anda. Setelah analisis pertama, IAM Access Analyzer menganalisis kebijakan ini secara berkala. Jika Anda menambahkan kebijakan baru atau mengubah kebijakan yang ada, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui dalam waktu sekitar 30 menit.

Saat menganalisis kebijakan, jika IAM Access Analyzer mengidentifikasi salah satu yang memberikan akses ke prinsipal eksternal yang tidak berada dalam zona kepercayaan Anda, itu menghasilkan temuan. Setiap temuan mencakup detail tentang sumber daya, entitas eksternal dengan akses ke sana, dan izin yang diberikan sehingga Anda dapat mengambil tindakan yang sesuai. Anda dapat melihat detail yang disertakan dalam temuan untuk menentukan apakah akses sumber daya disengaja atau merupakan potensi risiko yang harus Anda selesaikan. Saat Anda menambahkan kebijakan ke sumber daya, atau memperbarui kebijakan yang ada, IAM Access Analyzer menganalisis kebijakan tersebut. IAM Access Analyzer juga menganalisis semua kebijakan berbasis sumber daya secara berkala.

Pada kesempatan langka dalam kondisi tertentu, IAM Access Analyzer tidak menerima pemberitahuan tentang kebijakan yang ditambahkan atau diperbarui, yang dapat menyebabkan

penundaan dalam temuan yang dihasilkan. Penganalisis Akses IAM dapat memakan waktu hingga 6 jam untuk menghasilkan atau menyelesaikan temuan jika Anda membuat atau menghapus titik akses multi-wilayah yang terkait dengan bucket Amazon S3, atau memperbarui kebijakan untuk titik akses multi-wilayah. Selain itu, jika ada masalah pengiriman dengan perubahan pembatasan pengiriman AWS CloudTrail log atau kebijakan kontrol sumber daya (RCP), perubahan kebijakan tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan. Ketika ini terjadi, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya, yaitu dalam waktu 24 jam. Jika Anda ingin mengonfirmasi perubahan yang Anda buat pada kebijakan menyelesaikan masalah akses yang dilaporkan dalam temuan, Anda dapat memindai ulang sumber daya yang dilaporkan dalam temuan dengan menggunakan tautan Pindai ulang di halaman rin cian Temuan, atau dengan menggunakan pen `StartResourceScan` goperasian API Penganalisis Akses IAM. Untuk mempelajari selengkapnya, lihat Selesaikan temuan IAM Access Analyzer.

> **Important**
>
> Untuk akses eksternal, IAM Access Analyzer hanya menganalisis kebijakan yang diterapkan ke sumber daya di Wilayah yang sama AWS di mana ia diaktifkan. Untuk memantau semua sumber daya di AWS lingkungan Anda, Anda harus membuat penganalisis akses eksternal untuk mengaktifkan Penganalisis Akses IAM di setiap Wilayah tempat Anda menggunakan AWS sumber daya yang didukung.
>
> Untuk akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat penganalisis akses yang tidak digunakan di setiap Wilayah tempat Anda memiliki sumber daya tidak diperlukan.

IAM Access Analyzer menganalisis jenis sumber daya berikut untuk akses eksternal:

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- AWS Identity and Access Management peran
- AWS Key Management Service kunci
- AWS Lambda fungsi dan lapisan
- Antrean Amazon Simple Queue Service
- AWS Secrets Manager rahasia
- Topik Amazon Simple Notification Service

- Cuplikan volume Amazon Elastic Block Store
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Repositori Registri Wadah Elastis Amazon
- Sistem file Amazon Elastic File System
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Mengidentifikasi akses internal ke sumber daya penting bisnis

Untuk sumber daya penting bisnis pilihan Anda, IAM Access Analyzer membantu Anda mengidentifikasi prinsipal mana dalam organisasi atau akun Anda yang memiliki akses ke sumber daya tersebut. Analisis ini mendukung penerapan prinsip hak istimewa paling sedikit dengan memastikan bahwa sumber daya yang Anda tentukan hanya dapat diakses oleh prinsipal yang dituju dalam organisasi Anda.

Untuk petunjuk tentang cara memulai, lihatBuat penganalisis akses internal IAM Access Analyzer. Setelah Anda membuat penganalisis, Anda bisaKelola penganalisis akses internal IAM Access Analyzer.

Analisis akses internal membantu Anda:

- Menentukan pengguna atau peran IAM mana dalam akun atau organisasi Anda yang dapat mengakses sumber daya yang Anda tentukan
- Memahami jalur akses antara prinsipal dan sumber daya dalam lingkungan Anda AWS
- Verifikasi bahwa kontrol akses Anda berfungsi sebagaimana dimaksud
- Tinjau temuan untuk menentukan apakah akses dimaksudkan dan aman atau jika akses tidak diinginkan dan menghadirkan risiko keamanan
- Mengidentifikasi dan memperbaiki akses yang tidak diinginkan dalam organisasi Anda

IAM Access Analyzer menganalisis jenis sumber daya berikut untuk akses internal:

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon

- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Mengidentifikasi akses yang tidak digunakan yang diberikan kepada pengguna dan peran IAM

IAM Access Analyzer membantu Anda mengidentifikasi dan meninjau akses yang tidak digunakan di AWS organisasi dan akun Anda. IAM Access Analyzer terus memantau semua peran dan pengguna IAM di AWS organisasi dan akun Anda dan menghasilkan temuan untuk akses yang tidak digunakan. Temuan ini menyoroti peran yang tidak digunakan, kunci akses yang tidak digunakan untuk pengguna IAM, dan kata sandi yang tidak digunakan untuk pengguna IAM. Untuk peran dan pengguna IAM aktif, temuan memberikan visibilitas ke layanan dan tindakan yang tidak digunakan.

Untuk petunjuk tentang membuat penganalisis akses yang tidak digunakan dan mengelola temuannya, lihatMengelola IAM Access Analyzer penganalisis akses yang tidak digunakan.

IAM Access Analyzer meninjau informasi terakhir yang diakses untuk semua peran dalam AWS organisasi dan akun Anda untuk membantu Anda mengidentifikasi akses yang tidak digunakan. Informasi tindakan IAM yang diakses terakhir membantu Anda mengidentifikasi tindakan yang tidak digunakan untuk peran di Anda Akun AWS. Untuk informasi selengkapnya, lihat Memperbaiki izin dalam AWS menggunakan informasi yang terakhir diakses.

Temuan untuk penganalisis akses eksternal, internal, dan yang tidak digunakan diatur ke dalam dasbor ringkasan visual. Dasbor menyoroti AWS sumber daya Anda dan Akun AWS yang memiliki temuan terbanyak dan memberikan rincian temuan berdasarkan jenis. Untuk informasi selengkapnya tentang dasbor, lihat Melihat dasbor temuan Penganalisis Akses IAM.

Memvalidasi kebijakan terhadap AWS praktik terbaik

Anda dapat memvalidasi kebijakan terhadap tata bahasa kebijakan IAM dan praktik AWS terbaik menggunakan pemeriksaan kebijakan dasar yang disediakan oleh validasi kebijakan IAM Access Analyzer. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor kebijakan JSON di konsol IAM. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan

Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik AWS terbaik. Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan validasi kebijakan, lihatValidasi kebijakan dengan IAM Access Analyzer. Untuk daftar pemeriksaan kebijakan yang dilakukan IAM Access Analyzer, lihatMemeriksa untuk memvalidasi kebijakan.

Memvalidasi kebijakan terhadap standar keamanan yang Anda tentukan

Anda dapat memvalidasi kebijakan terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan kustom IAM Access Analyzer. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor kebijakan JSON di konsol IAM. Melalui konsol, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang sudah ada. Melalui AWS CLI dan AWS API, Anda juga dapat memeriksa tindakan IAM tertentu yang menurut Anda penting tidak diizinkan oleh kebijakan. Pemeriksaan ini menyoroti pernyataan kebijakan yang memberikan akses baru. Anda dapat memperbarui pernyataan kebijakan dan menjalankan kembali pemeriksaan hingga kebijakan tersebut sesuai dengan standar keamanan Anda. Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan pemeriksaan kebijakan khusus, lihatValidasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer.

Menghasilkan kebijakan

IAM Access Analyzer menganalisis AWS CloudTrail log Anda untuk mengidentifikasi tindakan dan layanan yang telah digunakan oleh entitas IAM (pengguna atau peran) dalam rentang tanggal yang Anda tentukan. Kemudian menghasilkan kebijakan IAM yang didasarkan pada aktivitas akses itu. Anda dapat menggunakan kebijakan yang dibuat untuk menyempurnakan izin entitas dengan melampirkannya ke pengguna atau peran IAM. Untuk mempelajari lebih lanjut tentang membuat kebijakan menggunakan IAM Access Analyzer, lihatPembuatan kebijakan Penganalisis Akses IAM.

Harga untuk IAM Access Analyzer

IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per penganalisis per bulan.

- Anda akan dikenakan biaya untuk setiap penganalisis akses yang tidak digunakan yang Anda buat.

- Membuat penganalisis akses yang tidak digunakan di beberapa Wilayah akan mengakibatkan Anda dikenakan biaya untuk setiap penganalisis.
- Service-linked peran tidak dianalisis untuk aktivitas akses yang tidak digunakan dan tidak termasuk dalam jumlah total peran IAM yang dianalisis.

IAM Access Analyzer mengenakan biaya untuk analisis akses internal berdasarkan jumlah sumber daya yang dipantau per penganalisis akses internal per bulan.

IAM Access Analyzer mengenakan biaya untuk pemeriksaan kebijakan khusus berdasarkan jumlah permintaan API yang dibuat ke IAM Access Analyzer untuk memeriksa akses baru.

Untuk daftar lengkap harga dan biaya untuk IAM Access Analyzer, lihat harga IAM Access Analyzer.

Untuk melihat tagihan Anda, buka Dasbor Manajemen Penagihan dan Biaya di konsol AWS Manajemen Penagihan dan Biaya. Tagihan Anda berisi tautan ke laporan penggunaan yang memberikan detail tentang tagihan Anda. Untuk mempelajari lebih lanjut tentang Akun AWS penagihan, lihat Panduan AWS Billing Pengguna

Jika Anda memiliki pertanyaan tentang pen AWS agihan, akun, dan acara, hubungi Dukungan.

Temuan IAM Access Analyzer

IAM Access Analyzer menghasilkan temuan untuk akses eksternal, akses internal, dan akses yang tidak digunakan di organisasi atau Anda Akun AWS .

Untuk akses eksternal, IAM Access Analyzer menghasilkan temuan untuk setiap contoh kebijakan berbasis sumber daya yang memberikan akses ke sumber daya dalam zona kepercayaan Anda kepada prinsipal yang tidak berada dalam zona kepercayaan Anda. Saat Anda membuat penganalisis akses eksternal, Anda memilih organisasi atau Akun AWS menganalisis. Setiap penanggung jawab dalam organisasi atau akun yang Anda pilih untuk penganalisis dianggap tepercaya. Karena penanggung jawab dalam organisasi atau akun yang sama dipercaya, sumber daya dan penanggung jawab di dalam organisasi atau akun yang terdiri dari zona kepercayaan untuk penganalisis. Setiap berbagi yang berada dalam zona kepercayaan dianggap aman, sehingga IAM Access Analyzer tidak menghasilkan temuan. Misalnya, jika Anda memilih sebuah organisasi sebagai zona kepercayaan untuk seorang penganalisis, semua sumber daya dan penanggung jawab di organisasi berada di dalam zona kepercayaan. Jika Anda memberikan izin ke bucket Amazon S3 di salah satu akun anggota organisasi Anda ke kepala sekolah di akun anggota organisasi lain, IAM

Access Analyzer tidak akan menghasilkan temuan. Tetapi jika Anda memberikan izin kepada kepala sekolah di akun yang bukan anggota organisasi, IAM Access Analyzer menghasilkan temuan.

Untuk akses internal, IAM Access Analyzer menghasilkan temuan ketika ada jalur akses yang mungkin antara peran IAM atau pengguna dalam organisasi Anda dan sumber daya yang Anda tentukan. Mirip dengan analisis akses eksternal, ruang lingkup yang Anda pilih (organisasi atau akun) menentukan apa yang dianggap internal. Jika Anda memilih organisasi sebagai cakupan, IAM Access Analyzer akan menghasilkan temuan untuk jalur akses antara prinsipal dan sumber daya dalam organisasi Anda. Jika Anda memilih akun, temuan akan dihasilkan untuk jalur akses dalam akun tertentu. IAM Access Analyzer menggunakan penalaran otomatis untuk mengevaluasi semua kebijakan IAM untuk memantau siapa yang memiliki akses ke sumber daya Anda.

Kombinasi temuan akses eksternal dan internal dengan zona kepercayaan yang sama memberikan analisis komprehensif dari semua kemungkinan akses ke sumber daya tertentu, baik dari dalam maupun di luar batas kepercayaan yang Anda tetapkan.

Untuk akses yang tidak digunakan, IAM Access Analyzer menghasilkan temuan untuk akses yang tidak terpakai yang diberikan di organisasi dan akun Anda AWS . Saat Anda membuat penganalisis akses yang tidak digunakan, IAM Access Analyzer terus memantau semua peran dan pengguna IAM di AWS organisasi dan akun Anda dan menghasilkan temuan untuk akses yang tidak digunakan. IAM Access Analyzer menghasilkan jenis temuan berikut untuk akses yang tidak digunakan:

- Peran yang tidak digunakan — Peran tanpa aktivitas akses dalam jendela penggunaan yang ditentukan.
- Kunci akses dan kata sandi pengguna IAM yang tidak digunakan — Kredensi milik pengguna IAM yang belum digunakan untuk mengakses Anda Akun AWS di jendela penggunaan yang ditentukan.
- Izin yang tidak digunakan — Izin tingkat layanan dan tingkat tindakan yang tidak digunakan oleh peran dalam jendela penggunaan yang ditentukan. IAM Access Analyzer menggunakan kebijakan berbasis identitas yang dilampirkan pada peran untuk menentukan layanan dan tindakan yang dapat diakses oleh peran tersebut. IAM Access Analyzer mendukung peninjauan izin yang tidak digunakan untuk semua izin tingkat layanan. Untuk daftar lengkap izin tingkat tindakan yang didukung untuk temuan akses yang tidak digunakan, lihat. Tindakan IAM terakhir mengakses layanan dan tindakan informasi

> **Note**
>
> IAM Access Analyzer menawarkan temuan akses eksternal secara gratis. Ada biaya untuk temuan akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna

yang dianalisis per penganalisis per bulan. Ada juga biaya untuk temuan akses internal berdasarkan jumlah AWS sumber daya yang dipantau per penganalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Topik

- Memahami cara kerja temuan IAM Access Analyzer
- Memulai dengan AWS Identity and Access Management Access Analyzer
- Melihat dasbor temuan Penganalisis Akses IAM
- Tinjau temuan IAM Access Analyzer
- Filter temuan IAM Access Analyzer
- Arsipkan temuan IAM Access Analyzer
- Selesaikan temuan IAM Access Analyzer
- Temuan kesalahan IAM Access Analyzer
- IAM Access Analyzer mendukung jenis sumber daya untuk akses eksternal dan internal
- Administrator yang didelegasikan untuk IAM Access Analyzer
- Aturan arsip
- Pemantauan AWS Identity and Access Management Access Analyzer dengan Amazon EventBridge
- Integrasikan IAM Access Analyzer dengan AWS Security Hub CSPM
- Mencatat panggilan API IAM Access Analyzer dengan AWS CloudTrail
- Kunci filter IAM Access Analyzer
- Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

Memahami cara kerja temuan IAM Access Analyzer

Topik ini menjelaskan konsep dan istilah yang digunakan di IAM Access Analyzer untuk membantu Anda menjadi terbiasa dengan cara IAM Access Analyzer memantau akses ke sumber daya Anda AWS .

Temuan akses eksternal

Temuan akses eksternal dibuat hanya sekali untuk setiap instance sumber daya yang dibagikan di luar zona kepercayaan Anda. Setiap kali kebijakan berbasis sumber daya dimodifikasi, IAM Access

Analyzer menganalisis kebijakan tersebut. Jika kebijakan yang diperbarui membagikan sumber daya yang sudah diidentifikasi dalam temuan, tetapi menggunakan izin atau ketentuan yang berbeda, temuan baru akan dihasilkan untuk contoh pembagian sumber daya tersebut. Perubahan pada kebijakan kontrol sumber daya yang memengaruhi pembatasan kebijakan kontrol sumber daya (RCP) juga menghasilkan temuan baru. IAM Access Analyzer juga mengevaluasi konfigurasi kontrol akses yang ditetapkan oleh kebijakan deklaratif. Jika akses dalam temuan pertama dihapus, temuan itu diperbarui ke status Tersel esaikan.

Status semua temuan tetap Aktif sampai Anda mengarsipkannya atau menghapus akses yang menghasilkan temuan. Saat Anda menghapus akses, status pencarian diperbarui ke Tersel esaikan.

> **Note**
>
> Mungkin diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi untuk IAM Access Analyzer untuk menganalisis sumber daya dan kemudian memperbarui temuan akses eksternal. Perubahan pada kebijakan kontrol sumber daya (RCP) tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan. IAM Access Analyzer menganalisis kebijakan baru atau diperbarui selama pemindaian berkala berikutnya, yaitu dalam waktu 24 jam.

Bagaimana IAM Access Analyzer menghasilkan temuan untuk akses eksternal

AWS Identity and Access Management Access Analyzer menggunakan teknologi yang disebut Zelkova untuk menganalisis kebijakan IAM dan mengidentifikasi akses eksternal ke sumber daya.

Zelkova menerjemahkan kebijakan IAM ke dalam pernyataan logis yang setara dan menjalankannya melalui serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan). IAM Access Analyzer menerapkan Zelkova berulang kali ke kebijakan, menggunakan kueri yang semakin spesifik untuk mengkarakterisasi jenis akses yang diizinkan kebijakan berdasarkan kontennya. Untuk informasi lebih lanjut tentang teori modulo kepuasan, lihat Teori Modulo Ke puasan.

Untuk penganalisis akses eksternal, IAM Access Analyzer tidak memeriksa log akses untuk menentukan apakah entitas eksternal benar-benar mengakses sumber daya dalam zona kepercayaan Anda. Sebaliknya, ini menghasilkan temuan ketika kebijakan berbasis sumber daya memungkinkan akses ke sumber daya, terlepas dari apakah sumber daya tersebut diakses oleh entitas eksternal.

Selain itu, IAM Access Analyzer tidak mempertimbangkan status akun eksternal apa pun saat membuat penentuannya. Jika ini menunjukkan bahwa akun 111122223333 dapat mengakses bucket Amazon S3 Anda, akun tersebut tidak memiliki informasi apa pun tentang pengguna, peran, kebijakan kontrol layanan (SCP), atau konfigurasi relevan lainnya di akun tersebut. Ini untuk privasi pelanggan, karena IAM Access Analyzer tidak tahu siapa yang memiliki akun lain. Ini juga untuk keamanan, karena penting untuk mengetahui tentang akses eksternal potensial meskipun saat ini tidak ada prinsipal aktif yang dapat menggunakannya.

IAM Access Analyzer hanya mempertimbangkan kunci kondisi IAM tertentu yang tidak dapat dipengaruhi oleh pengguna eksternal secara langsung atau yang berdampak pada otorisasi. Untuk contoh kunci kondisi yang dipertimbangkan IAM Access Analyzer, lihat kunci filter Penganalisis Akses IAM.

IAM Access Analyzer saat ini tidak melaporkan temuan dari kepala Layanan AWS sekolah atau akun layanan internal. Dalam kasus yang jarang terjadi di mana ia tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses ke entitas eksternal, itu salah di sisi menyatakan temuan positif palsu. Ini karena IAM Access Analyzer dirancang untuk memberikan tampilan komprehensif tentang berbagi sumber daya di akun Anda dan untuk meminimalkan negatif palsu.

Temuan akses internal

Untuk menggunakan analisis akses internal, Anda harus terlebih dahulu mengonfigurasi penganalisis dengan memilih sumber daya tertentu yang ingin Anda pantau. Setelah dikonfigurasi, temuan akses internal dihasilkan ketika kepala sekolah (pengguna atau peran IAM) dalam organisasi atau akun Anda memiliki akses ke sumber daya yang Anda pilih. Temuan baru dihasilkan saat berikutnya penganalisis memindai sumber daya yang ditentukan dan mengidentifikasi prinsipal yang memiliki akses ke sumber daya. Jika kebijakan yang diperbarui mengizinkan prinsipal yang sudah diidentifikasi dalam temuan, tetapi dengan izin atau kondisi yang berbeda, temuan baru dihasilkan untuk contoh prinsip dan sumber daya tersebut. Kebijakan yang diperbarui ini bisa berupa kebijakan berbasis sumber daya, kebijakan berbasis identitas, kebijakan kontrol layanan (SCP), atau kebijakan kontrol sumber daya (RCP). IAM Access Analyzer juga mengevaluasi konfigurasi kontrol akses yang ditetapkan oleh kebijakan deklaratif.

> **Note**
>
> Temuan akses internal hanya tersedia menggunakan tindakan API ListFindings V2.

Bagaimana IAM Access Analyzer menghasilkan temuan untuk akses internal

Untuk menganalisis akses internal, Anda harus membuat penganalisis terpisah untuk temuan akses internal untuk sumber daya Anda, bahkan jika Anda telah membuat penganalisis untuk menghasilkan temuan akses eksternal atau temuan akses yang tidak digunakan.

Setelah membuat penganalisis akses internal, IAM Access Analyzer mengevaluasi semua kebijakan berbasis sumber daya, kebijakan berbasis identitas, kebijakan kontrol layanan (SCP), kebijakan kontrol sumber daya (RCP), dan batas izin dalam akun atau organisasi yang Anda tentukan.

Dengan membuat penganalisis yang didedikasikan untuk akses internal ke sumber daya pilihan Anda, Anda dapat mengidentifikasi:

- Ketika seorang kepala sekolah di organisasi atau akun Anda dapat mengakses sumber daya yang Anda pilih
- Total izin efektif yang diizinkan untuk prinsipal berdasarkan persimpangan semua kebijakan yang berlaku
- Jalur akses yang kompleks di mana prinsipal memperoleh akses berdasarkan kombinasi kebijakan identitas dan kebijakan sumber daya

> **Note**
>
> IAM Access Analyzer tidak dapat menghasilkan temuan akses internal untuk organisasi yang berisi lebih dari 70.000 prinsipal (pengguna dan peran IAM digabungkan).

Temuan akses yang tidak digunakan

Temuan akses yang tidak digunakan dihasilkan untuk entitas IAM (prinsipal) dalam akun atau organisasi yang dipilih berdasarkan jumlah hari yang ditentukan saat membuat penganalisis. Temuan baru dihasilkan saat penganalisis memindai entitas berikutnya jika salah satu kondisi berikut terpenuhi:

- Peran tidak aktif untuk jumlah hari yang ditentukan.
- Izin yang tidak digunakan, kata sandi pengguna yang tidak digunakan, atau kunci akses pengguna yang tidak digunakan melampaui jumlah hari yang ditentukan.

> **Note**
>
> Temuan akses yang tidak digunakan hanya tersedia menggunakan tindakan API ListFindings V2.

Bagaimana IAM Access Analyzer menghasilkan temuan untuk akses yang tidak digunakan

Untuk menganalisis akses yang tidak digunakan, Anda harus membuat penganalisis terpisah untuk temuan akses yang tidak digunakan untuk peran Anda, bahkan jika Anda telah membuat penganalisis untuk menghasilkan temuan akses eksternal atau internal untuk sumber daya Anda.

Setelah membuat penganalisis akses yang tidak digunakan, IAM Access Analyzer meninjau aktivitas akses untuk mengidentifikasi akses yang tidak digunakan. IAM Access Analyzer memeriksa informasi terakhir yang diakses untuk semua pengguna IAM, peran IAM termasuk peran layanan, kunci akses pengguna, dan kata sandi pengguna di seluruh AWS organisasi dan akun Anda. Ini membantu Anda mengidentifikasi akses yang tidak digunakan.

> **Note**
>
> Per an terkait layanan adalah jenis peran layanan khusus yang terkait dengan Layanan AWS dan dimiliki oleh layanan. Service-linked peran tidak dianalisis oleh penganalisis akses yang tidak digunakan.

Untuk peran dan pengguna IAM aktif, IAM Access Analyzer menggunakan informasi terakhir yang diakses untuk layanan dan tindakan IAM untuk mengidentifikasi izin yang tidak digunakan. Ini memungkinkan Anda untuk menskalakan proses peninjauan Anda di tingkat AWS organisasi dan akun. Anda juga dapat menggunakan informasi tindakan yang terakhir diakses untuk penyelidikan lebih dalam tentang peran individu. Ini memberikan wawasan yang lebih terperinci tentang izin spesifik mana yang tidak digunakan.

Dengan membuat penganalisis yang didedikasikan untuk akses yang tidak digunakan, Anda dapat meninjau dan mengidentifikasi akses yang tidak digunakan secara komprehensif di seluruh AWS lingkungan Anda, melengkapi temuan yang dihasilkan oleh penganalisis akses eksternal yang ada.

Memulai dengan AWS Identity and Access Management Access Analyzer

Gunakan informasi dalam topik ini untuk mempelajari tentang persyaratan yang diperlukan untuk menggunakan dan mengelola AWS Identity and Access Management Access Analyzer.

Jika Anda menggunakan IAM Access Analyzer dalam suatu AWS Organizations organisasi, Anda bisaTambahkan administrator yang didelegasikan untuk IAM Access Analyzer. Untuk informasi tentang batas layanan, lihatKuota Penganalisis Akses IAM.

Izin diperlukan untuk menggunakan IAM Access Analyzer

Agar berhasil mengkonfigurasi dan menggunakan IAM Access Analyzer, akun yang Anda gunakan harus diberikan izin yang diperlukan.

AWS kebijakan terkelola untuk IAM Access Analyzer

AWS Identity and Access Management Access Analyzer menyediakan AWS kebijakan terkelola untuk membantu Anda memulai dengan cepat.

- IAMAccessAnalyzerFullAccess- Memungkinkan akses penuh ke IAM Access Analyzer untuk administrator. Kebijakan ini juga memungkinkan pembuatan peran terkait layanan yang diperlukan agar IAM Access Analyzer dapat menganalisis sumber daya di akun atau organisasi Anda. AWS
- IAMAccessAnalyzerReadOnlyAccess- Memungkinkan akses read-only ke IAM Access Analyzer. Anda harus menambahkan kebijakan tambahan ke identitas IAM Anda (pengguna, grup pengguna, atau peran) agar mereka dapat melihat temuan mereka.

Sumber daya yang ditentukan oleh IAM Access Analyzer

Untuk melihat sumber daya yang ditentukan oleh IAM Access Analyzer, lihat Jenis sumber daya yang ditentukan oleh IAM Access Analyzer di Referensi Otorisasi Layanan.

Diperlukan izin layanan Penganalisis Akses IAM

IAM Access Analyzer menggunakan peran terkait layanan (SLR) bernama. `AWSServiceRoleForAccessAnalyzer` SLR ini memberikan layanan akses read-only untuk menganalisis AWS sumber daya dengan kebijakan berbasis sumber daya dan menganalisis akses yang tidak digunakan atas nama Anda. Layanan membuat peran di akun Anda dalam skenario berikut:

- Anda membuat penganalisis akses eksternal dengan akun Anda sebagai zona kepercayaan.

- Anda membuat penganalisis akses yang tidak digunakan dengan akun Anda sebagai akun yang dipilih.
- Anda membuat penganalisis akses internal dengan akun Anda sebagai zona kepercayaan.

Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer.

> **Note**
>
> Penganalisis Akses IAM adalah Regional. Untuk akses eksternal dan internal, Anda harus mengaktifkan Penganalisis Akses IAM di setiap Wilayah secara independen.
> Untuk akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat penganalisis di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

Dalam beberapa kasus, setelah Anda membuat penganalisis di IAM Access Analyzer, halaman Temuan atau dasbor dimuat tanpa temuan atau ringkasan. Hal ini mungkin disebabkan oleh penundaan pada konsol untuk mengisi temuan Anda. Anda mungkin perlu menyegarkan browser secara manual atau memeriksa kembali nanti untuk melihat temuan atau ringkasan Anda. Jika Anda masih tidak melihat temuan apa pun untuk penganalisis akses eksternal, itu karena Anda tidak memiliki sumber daya yang didukung di akun Anda yang dapat diakses oleh entitas eksternal. Jika kebijakan yang memberikan akses ke entitas eksternal diterapkan ke sumber daya, IAM Access Analyzer menghasilkan temuan.

> **Note**
>
> Untuk penganalisis akses eksternal, mungkin diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi agar IAM Access Analyzer menganalisis sumber daya dan kemudian menghasilkan temuan baru atau memperbarui temuan yang ada untuk akses ke sumber daya.
> Saat Anda membuat penganalisis akses internal, mungkin diperlukan beberapa menit atau jam sebelum temuan tersedia. Setelah pemindaian awal, IAM Access Analyzer secara otomatis memindai ulang semua kebijakan setiap 24 jam.
> Untuk semua jenis penganalisis akses, pembaruan untuk temuan mungkin tidak segera tercermin di dasbor.

Diperlukan izin Penganalisis Akses IAM untuk melihat dasbor temuan

Untuk melihat dasbor temuan Penganalisis Akses IAM, akun yang Anda gunakan harus diberikan akses untuk melakukan tindakan yang diperlukan berikut:

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Untuk melihat semua tindakan yang ditentukan oleh IAM Access Analyzer, lihat Tindakan yang ditentukan oleh Penganalisis Akses IAM di Referensi Otorisasi Layanan.

Status Penganalisis Akses IAM

Untuk melihat status analyzer Anda, pilih Penganalisis. Penganalisis yang dibuat untuk organisasi atau akun dapat memiliki status sebagai berikut:

Status	Deskripsi
Aktif	Untuk penganalisis akses eksternal dan internal, penganalisis secara aktif memantau sumber daya dalam zona kepercayaannya. Analyzer secara aktif menghasilkan temuan baru dan memperbarui temuan yang ada. Untuk penganalisis akses yang tidak digunakan , penganalisis secara aktif memantau akses yang tidak digunakan dalam organisasi yang dipilih atau Akun AWS dalam periode pelacakan yang ditentukan. Analyzer secara aktif menghasilkan temuan baru dan memperbarui temuan yang ada.
Membuat	Pembuatan penganalisis masih dalam proses. Penganalisis menjadi aktif setelah pembuatan selesai.

Status	Deskripsi
Nonaktif	Penganalisis dinonaktifkan karena tindakan yang diambil oleh AWS Organizations administrator. Misalnya, menghapus akun penganalisis sebagai administrator yang telah didedikasikan untuk Penganalisis Akses IAM. Ketika penganalisis dalam keadaan dinonaktifkan, itu tidak menghasilkan temuan baru atau memperbarui temuan yang ada.
Gagal	Pembuatan penganalisis gagal karena masalah konfigurasi. Penganalisis tidak akan menghasilkan temuan apa pun. Hapus penganalisis dan buat penganalisis baru.

Buat penganalisis akses eksternal IAM Access Analyzer

Untuk mengaktifkan penganalisis akses eksternal di Wilayah, Anda harus membuat penganalisis di Wilayah tersebut. Anda harus membuat penganalisis akses eksternal di setiap Wilayah tempat Anda ingin memantau akses ke sumber daya Anda.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia.

Buat penganalisis akses eksternal dengan Akun AWS sebagai zona kepercayaan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis sumber daya - Akses eksternal.
5. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
6. Masukkan nama penganalisis.

7. Pilih Akun saat ini sebagai zona kepercayaan untuk penganalisis.

> **Note**
>
> Jika akun Anda bukan akun AWS Organizations manajemen atau akun administrator yang didelegasikan, Anda hanya dapat membuat satu penganalisis dengan akun Anda sebagai zona kepercayaan.

8. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
9. Pilih Membuat penganalisis.

Saat Anda membuat penganalisis akses eksternal untuk mengaktifkan IAM Access Analyzer, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Buat penganalisis akses eksternal dengan organisasi sebagai zona kepercayaan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis sumber daya - Akses eksternal.
5. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
6. Masukkan nama penganalisis.
7. Pilih Organisasi saat ini sebagai zona kepercayaan untuk penganalisis.
8. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
9. Pilih Kirim.

Saat Anda membuat penganalisis akses eksternal dengan organisasi sebagai zona kepercayaan, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di setiap akun organisasi Anda.

Kelola penganalisis akses eksternal IAM Access Analyzer

Untuk mengaktifkan penganalisis akses eksternal di Wilayah, Anda harus membuat penganalisis di Wilayah tersebut. Anda harus membuat penganalisis akses eksternal di setiap Wilayah tempat Anda ingin memantau akses ke sumber daya Anda.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia.

Perbarui penganalisis akses eksternal

Gunakan prosedur berikut untuk memperbarui penganalisis akses eksternal.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih nama penganalisis akses eksternal untuk dikelola.
4. Pada tab Aturan arsip, Anda dapat membuat, mengedit, atau menghapus aturan arsip untuk penganalisis. Untuk informasi selengkapnya, lihat Aturan arsip.
5. Pada tab Tag, Anda dapat mengelola dan membuat tag untuk penganalisis. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.

Hapus penganalisis akses eksternal

Gunakan prosedur berikut untuk menghapus penganalisis akses eksternal. Saat Anda menghapus penganalisis, sumber daya tidak lagi dipantau dan tidak ada temuan baru yang dihasilkan. Semua temuan yang dihasilkan oleh penganalisis dihapus.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih nama penganalisis akses eksternal untuk dihapus.
4. Pilih Hapus penganalisis.
5. Masukkan hapus dan pilih Hapus untuk mengonfirmasi penghapusan penganalisis.

Buat penganalisis akses internal IAM Access Analyzer

Untuk mengaktifkan penganalisis akses internal di Wilayah, Anda harus membuat penganalisis di Wilayah tersebut. Anda harus membuat penganalisis akses internal di setiap Wilayah tempat Anda ingin memantau akses ke sumber daya Anda.

IAM Access Analyzer mengenakan biaya untuk analisis akses internal berdasarkan jumlah sumber daya yang dipantau per penganalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia. IAM Access Analyzer tidak dapat menghasilkan temuan akses internal untuk organisasi yang berisi lebih dari 70.000 prinsipal (pengguna dan peran IAM digabungkan).
> Anda hanya dapat membuat satu penganalisis akses internal tingkat organisasi dalam suatu organisasi. AWS

Buat penganalisis akses internal dengan Akun AWS sebagai zona kepercayaan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis sumber daya - Akses internal.
5. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
6. Masukkan nama penganalisis.
7. Pilih Akun saat ini sebagai zona kepercayaan untuk penganalisis.

   > **Note**
   >
   > Jika akun Anda bukan akun AWS Organizations manajemen atau akun administrator yang didelegasikan, Anda hanya dapat membuat satu penganalisis dengan akun Anda sebagai zona kepercayaan.

8. Di bagian Sumber Daya untuk menganalisis, tambahkan sumber daya untuk dipantau oleh penganalisis.

   - Untuk menambahkan sumber daya berdasarkan akun, pilih Tambah > Tambahkan sumber daya dari akun yang dipilih.
     1. Pilih Semua jenis sumber daya yang didukung atau pilih Tentukan jenis sumber daya tertentu dan pilih jenis sumber daya dari daftar Jenis sumber daya.

Penganalisis akses internal mendukung jenis sumber daya berikut:

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

2. Pilih Tambahkan sumber daya.

- Untuk menambahkan sumber daya berdasarkan Nama Sumber Daya Amazon (ARN), pilih Tambahkan sumber daya > Tambahkan sumber daya dengan menempelkan ARN sumber daya.

> **Note**
>
> ARNs harus sama persis — wildcard tidak didukung. Untuk Amazon S3, hanya bucket yang ARNs didukung. Objek ARNs dan awalan Amazon S3 tidak didukung.

1. Untuk setiap ARN sumber daya, masukkan ID pemilik akun dan ARN sumber daya dipisahkan dengan koma. Masukkan satu ID pemilik akun dan sumber daya ARN per baris.
2. Pilih Tambahkan sumber daya.

- Untuk menambahkan sumber daya dengan file CSV, pilih Tambahkan sumber daya > Tambahkan sumber daya dengan mengunggah CSV.

  Anda dapat menggunakan Penjelajah Sumber Daya AWSuntuk mencari sumber daya di akun Anda dan mengekspor file CSV. Kemudian Anda dapat mengunggah file CSV untuk mengonfigurasi sumber daya untuk dipantau oleh penganalisis.

  1. Pilih Pilih file dan pilih file CSV dari komputer Anda.
  2. Pilih Tambahkan sumber daya.

9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Membuat penganalisis.

Saat Anda membuat penganalisis akses internal untuk mengaktifkan IAM Access Analyzer, nama peran terkait layanan akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Buat penganalisis akses internal dengan organisasi sebagai zona kepercayaan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis sumber daya - Akses internal.
5. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
6. Masukkan nama penganalisis.
7. Pilih Seluruh organisasi sebagai zona kepercayaan untuk penganalisis.
8. Di bagian Sumber Daya untuk menganalisis, tambahkan sumber daya untuk dipantau oleh penganalisis.

   - Untuk menambahkan sumber daya untuk akun, pilih Tambahkan sumber daya > Tambahkan sumber daya dari akun yang dipilih.
     1. Pilih Semua jenis sumber daya yang didukung atau pilih Tentukan jenis sumber daya tertentu dan pilih jenis sumber daya dari daftar Jenis sumber daya.

        Penganalisis akses internal mendukung jenis sumber daya berikut:

        - Bucket Amazon Simple Storage Service
        - Ember direktori Layanan Penyimpanan Sederhana Amazon
        - Cuplikan DB Layanan Amazon Relational Database Service
        - Cuplikan cluster DB Layanan Relational Database Service Amazon
        - Aliran Amazon DynamoDB
        - Tabel Amazon DynamoDB
     2. Untuk memilih akun dari organisasi Anda, pilih Pilih dari organisasi. Di bagian Pilih akun, pilih Hierarki untuk memilih akun berdasarkan struktur organisasi atau Daftar untuk memilih akun dari daftar semua akun di organisasi Anda.

        Untuk memasukkan akun secara manual dari organisasi Anda, pilih Masukkan ID AWS akun. Masukkan satu atau lebih Akun AWS IDs dipisahkan dengan koma di bidang ID AWS akun.

3. Pilih Tambahkan sumber daya.

- Untuk menambahkan sumber daya berdasarkan Nama Sumber Daya Amazon (ARN), pilih Tambahkan sumber daya > Tambahkan sumber daya dengan menempelkan ARN sumber daya.

  > **Note**
  >
  > ARNs harus sama persis — wildcard tidak didukung. Untuk Amazon S3, hanya bucket yang ARNs didukung. Objek ARNs dan awalan Amazon S3 tidak didukung.

  1. Untuk setiap ARN sumber daya, masukkan ID pemilik akun dan ARN sumber daya dipisahkan dengan koma. Masukkan satu ID pemilik akun dan sumber daya ARN per baris.
  2. Pilih Tambahkan sumber daya.

- Untuk menambahkan sumber daya dengan file CSV, pilih Tambahkan sumber daya > Tambahkan sumber daya dengan mengunggah CSV.

  Anda dapat menggunakan Penjelajah Sumber Daya AWSuntuk mencari sumber daya di akun Anda dan mengekspor file CSV. Kemudian Anda dapat mengunggah file CSV untuk mengonfigurasi sumber daya untuk dipantau oleh penganalisis.

  1. Pilih Pilih file dan pilih file CSV dari komputer Anda.
  2. Pilih Tambahkan sumber daya.

9. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
10. Pilih Kirim.

Saat Anda membuat penganalisis akses internal dengan organisasi sebagai zona kepercayaan, nama peran terkait layanan `AWSServiceRoleForAccessAnalyzer` dibuat di setiap akun organisasi Anda.

Kelola penganalisis akses internal IAM Access Analyzer

Untuk mengaktifkan penganalisis akses internal di Wilayah, Anda harus membuat penganalisis di Wilayah tersebut. Anda harus membuat penganalisis akses internal di setiap Wilayah tempat Anda ingin memantau akses ke sumber daya Anda.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia.

Perbarui penganalisis akses internal

Gunakan prosedur berikut untuk memperbarui penganalisis akses internal.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih nama penganalisis akses internal untuk dikelola.
4. Pada tab Aturan arsip, Anda dapat membuat, mengedit, atau menghapus aturan arsip untuk penganalisis. Untuk informasi selengkapnya, lihat Aturan arsip.
5. Pada tab Tag, Anda dapat mengelola dan membuat tag untuk penganalisis. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.
6. Pada tab Sumber Daya, pilih Edit di bagian Sumber Daya untuk menganalisis.
   a. Untuk menambahkan sumber daya berdasarkan akun, pilih Tambahkan sumber daya > Tambahkan sumber daya dari akun yang dipilih.
      1. Pilih Semua jenis sumber daya yang didukung atau pilih Tentukan jenis sumber daya tertentu dan pilih jenis sumber daya dari daftar Jenis sumber daya.

         Penganalisis akses internal mendukung jenis sumber daya berikut:

         - Bucket Amazon Simple Storage Service
         - Ember direktori Layanan Penyimpanan Sederhana Amazon
         - Cuplikan DB Layanan Amazon Relational Database Service
         - Cuplikan cluster DB Layanan Relational Database Service Amazon
         - Aliran Amazon DynamoDB
         - Tabel Amazon DynamoDB

      2. Pilih Tambahkan sumber daya.
   b. Untuk menambahkan sumber daya berdasarkan Nama Sumber Daya Amazon (ARN), pilih Tambahkan sumber daya > Tambahkan sumber daya dengan menempelkan ARN sumber daya.

> **Note**
>
> ARNs harus sama persis — wildcard tidak didukung. Untuk Amazon S3, hanya bucket yang ARNs didukung. Objek ARNs dan awalan Amazon S3 tidak didukung.

1. Untuk setiap ARN sumber daya, masukkan ID pemilik akun dan ARN sumber daya dipisahkan dengan koma. Masukkan satu ID pemilik akun dan sumber daya ARN per baris.
2. Pilih Tambahkan sumber daya.

c. Untuk menambahkan sumber daya dengan file CSV, pilih Tambahkan sumber daya > Tambahkan sumber daya dengan mengunggah CSV.

Anda dapat menggunakan Penjelajah Sumber Daya AWSuntuk mencari sumber daya di akun Anda dan mengekspor file CSV. Kemudian Anda dapat mengunggah file CSV untuk mengonfigurasi sumber daya untuk dipantau oleh penganalisis.

1. Pilih Pilih file dan pilih file CSV dari komputer Anda.
2. Pilih Tambahkan sumber daya.

d. Untuk menghapus sumber daya dari penganalisis, pilih kotak centang di sebelah sumber daya yang akan dihapus dan pilih Hapus.

e. Pilih Simpan perubahan.

> **Note**
>
> Setiap pembaruan untuk penganalisis akan dievaluasi pada pemindaian ulang otomatis berikutnya dalam waktu 24 jam.

Hapus penganalisis akses internal

Gunakan prosedur berikut untuk menghapus penganalisis akses internal. Saat Anda menghapus penganalisis, sumber daya tidak lagi dipantau dan tidak ada temuan baru yang dihasilkan. Semua temuan yang dihasilkan oleh penganalisis dihapus.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih nama penganalisis akses internal untuk dihapus.
4. Pilih Hapus penganalisis.
5. Masukkan hapus dan pilih Hapus untuk mengonfirmasi penghapusan penganalisis.

Buat IAM Access Analyzer penganalisis akses yang tidak digunakan

Buat penganalisis akses yang tidak digunakan untuk akun saat ini

Gunakan prosedur berikut untuk membuat penganalisis akses yang tidak terpakai untuk satu. Akun AWS Untuk akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat analyzer di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per bulan per penganalisis. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis utama - Akses yang tidak digunakan.
5. Masukkan nama penganalisis.
6. Untuk periode Tracking, masukkan jumlah hari untuk analisis. Penganalisis hanya akan mengevaluasi izin untuk entitas IAM dalam akun yang dipilih yang telah ada selama seluruh periode pelacakan. Misalnya, jika Anda menetapkan periode pelacakan 90 hari, hanya izin yang berusia minimal 90 hari yang akan dianalisis, dan temuan akan dihasilkan jika tidak menunjukkan penggunaan selama periode ini. Anda dapat memasukkan nilai antara 1 dan 365 hari.
7. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.

8. Untuk Akun yang dipilih, pilih Akun saat ini.

> **Note**
>
> Jika akun Anda bukan akun AWS Organizations manajemen atau akun administrator yang didelegasikan, Anda hanya dapat membuat satu penganalisis dengan akun Anda sebagai akun yang dipilih.

9. Tidak wajib. Di bagian Kecualikan pengguna dan peran IAM dengan tag, Anda dapat menentukan pasangan nilai kunci untuk pengguna IAM dan peran yang akan dikecualikan dari analisis akses yang tidak digunakan. Temuan tidak akan dihasilkan untuk pengguna IAM yang dikecualikan dan peran yang cocok dengan pasangan nilai kunci. Untuk tombol Tag, masukkan nilai yang panjangnya 1 hingga 128 karakter dan tidak diawali dengan`aws:`. Untuk Nilai, Anda dapat memasukkan nilai yang panjangnya 0 hingga 256 karakter. Jika Anda tidak memasukkan Nilai, aturan diterapkan ke semua prinsipal dengan kunci Tag yang ditentukan. Pilih Tambahkan pengecualian baru untuk menambahkan pasangan nilai kunci tambahan untuk dikecualikan.
10. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
11. Pilih Membuat penganalisis.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, peran terkait layanan bernama akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Buat penganalisis akses yang tidak digunakan dengan organisasi saat ini

Gunakan prosedur berikut untuk membuat penganalisis akses yang tidak terpakai bagi organisasi untuk meninjau secara terpusat semua Akun AWS dalam organisasi. Untuk analisis akses yang tidak digunakan, temuan untuk penganalisis tidak berubah berdasarkan Wilayah. Membuat analyzer di setiap Wilayah di mana Anda memiliki sumber daya tidak diperlukan.

IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per bulan per penganalisis. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

> **Note**
>
> Jika akun anggota dihapus dari organisasi, penganalisis akses yang tidak digunakan akan berhenti menghasilkan temuan baru dan memperbarui temuan yang ada untuk akun tersebut

setelah 24 jam. Temuan yang terkait dengan akun anggota yang dihapus dari organisasi akan dihapus secara permanen setelah 90 hari.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Pilih Membuat penganalisis.
4. Di bagian Analisis, pilih Analisis utama - Akses yang tidak digunakan.
5. Masukkan nama penganalisis.
6. Untuk periode Tracking, masukkan jumlah hari untuk analisis. Penganalisis hanya akan mengevaluasi izin untuk entitas IAM dalam akun organisasi yang dipilih yang telah ada selama seluruh periode pelacakan. Misalnya, jika Anda menetapkan periode pelacakan 90 hari, hanya izin yang berusia minimal 90 hari yang akan dianalisis, dan temuan akan dihasilkan jika tidak menunjukkan penggunaan selama periode ini. Anda dapat memasukkan nilai antara 1 dan 365 hari.
7. Di bagian Detail Analyzer, konfirmasikan bahwa Wilayah yang ditampilkan adalah Wilayah tempat Anda ingin mengaktifkan IAM Access Analyzer.
8. Untuk Akun yang dipilih, pilih Organisasi saat ini.
9. Tidak wajib. Di bagian Kecualikan Akun AWS dari analisis, Anda dapat memilih Akun AWS di organisasi Anda untuk mengecualikan dari analisis akses yang tidak digunakan. Temuan tidak akan dihasilkan untuk akun yang dikecualikan.

    a. Untuk menentukan akun individu yang IDs akan dikecualikan, pilih Tentukan Akun AWS ID dan masukkan akun yang IDs dipisahkan dengan koma di bidang Akun AWS ID. Pilih Kecualikan. Akun tersebut kemudian tercantum dalam tabel Akun AWS untuk mengecualikan.

    b. Untuk memilih dari daftar akun di organisasi yang akan dikecualikan, pilih Pilih dari organisasi.

        i. Anda dapat mencari akun berdasarkan nama, email, dan ID akun di bidang Kecualikan akun dari organisasi.

        ii. Pilih Hierarki untuk melihat akun Anda berdasarkan unit organisasi atau pilih Daftar untuk melihat daftar semua akun individual di organisasi Anda.

        iii. Pilih Kecualikan semua akun saat ini untuk mengecualikan semua akun dalam unit organisasi atau pilih Kecualikan untuk mengecualikan akun individual.

Akun tersebut kemudian tercantum dalam tabel Akun AWS untuk mengecualikan.

> **Note**
>
> Akun yang dikecualikan tidak dapat menyertakan akun pemilik penganalisis organisasi. Ketika akun baru ditambahkan ke organisasi Anda, mereka tidak dikecualikan dari analisis, bahkan jika Anda sebelumnya mengecualikan semua akun saat ini dalam unit organisasi. Untuk informasi selengkapnya tentang mengecualikan akun setelah membuat penganalisis akses yang tidak digunakan, lihat. Mengelola IAM Access Analyzer penganalisis akses yang tidak digunakan

10. Tidak wajib. Di bagian Kecualikan pengguna dan peran IAM dengan tag, Anda dapat menentukan pasangan nilai kunci untuk pengguna IAM dan peran yang akan dikecualikan dari analisis akses yang tidak digunakan. Temuan tidak akan dihasilkan untuk pengguna IAM yang dikecualikan dan peran yang cocok dengan pasangan nilai kunci. Untuk tombol Tag, masukkan nilai yang panjangnya 1 hingga 128 karakter dan tidak diawali dengan`aws:`. Untuk Nilai, Anda dapat memasukkan nilai yang panjangnya 0 hingga 256 karakter. Jika Anda tidak memasukkan Nilai, aturan diterapkan ke semua prinsipal dengan kunci Tag yang ditentukan. Pilih Tambahkan pengecualian baru untuk menambahkan pasangan nilai kunci tambahan untuk dikecualikan.
11. Tidak wajib. Tambahkan tanda yang ingin Anda terapkan ke penganalisis.
12. Pilih Membuat penganalisis.

Saat Anda membuat penganalisis akses yang tidak digunakan untuk mengaktifkan IAM Access Analyzer, peran terkait layanan bernama akan `AWSServiceRoleForAccessAnalyzer` dibuat di akun Anda.

Mengelola IAM Access Analyzer penganalisis akses yang tidak digunakan

Gunakan informasi dalam topik ini untuk mempelajari cara memperbarui atau menghapus penganalisis akses yang tidak terpakai yang sudah ada.

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, perlu waktu agar temuan tersedia.

Perbarui penganalisis akses yang tidak digunakan

Gunakan prosedur berikut untuk memperbarui penganalisis akses yang tidak digunakan.

IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per bulan per penganalisis. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih nama penganalisis akses yang tidak terpakai untuk dikelola.
4. Pada tab Pengecualian, jika penganalisis dibuat untuk organisasi sebagai ruang lingkup analisis, pilih Kelola di bagian Dikecualikan Akun AWS.

    a. Untuk menentukan akun individu yang IDs akan dikecualikan, pilih Tentukan Akun AWS ID dan masukkan akun yang IDs dipisahkan dengan koma di bidang Akun AWS ID. Pilih Kecualikan. Akun tersebut kemudian tercantum dalam tabel Akun AWS untuk mengecualikan.

    b. Untuk memilih dari daftar akun di organisasi yang akan dikecualikan, pilih Pilih dari organisasi.

        i. Anda dapat mencari akun berdasarkan nama, email, dan ID akun di bidang Kecualikan akun dari organisasi.

        ii. Pilih Hierarki untuk melihat akun Anda berdasarkan unit organisasi atau pilih Daftar untuk melihat daftar semua akun individual di organisasi Anda.

        iii. Pilih Kecualikan semua akun saat ini untuk mengecualikan semua akun dalam unit organisasi atau pilih Kecualikan untuk mengecualikan akun individual.

        Akun tersebut kemudian tercantum dalam tabel Akun AWS untuk mengecualikan.

    c. Untuk menghapus akun yang akan dikecualikan, pilih Hapus di sebelah akun di tabel Akun AWS untuk mengecualikan.

    d. Pilih Simpan perubahan.

> **Note**
>
> - Akun yang dikecualikan tidak dapat menyertakan akun pemilik penganalisis organisasi.
> - Ketika akun baru ditambahkan ke organisasi Anda, mereka tidak dikecualikan dari analisis, bahkan jika Anda sebelumnya mengecualikan semua akun saat ini dalam unit organisasi.
> - Setelah Anda memperbarui pengecualian untuk penganalisis, diperlukan waktu hingga dua hari untuk daftar akun yang dikecualikan diperbarui.

5. Pada tab Pengecualian, pilih Kelola di bagian Pengguna dan peran IAM yang Dikecualikan dengan tag.

   a. Anda dapat menentukan pasangan nilai kunci untuk pengguna IAM dan peran yang akan dikecualikan dari analisis akses yang tidak digunakan. Untuk tombol Tag, masukkan nilai yang panjangnya 1 hingga 128 karakter dan tidak diawali dengan`aws:`. Untuk Nilai, Anda dapat memasukkan nilai yang panjangnya 0 hingga 256 karakter. Jika Anda tidak memasukkan Nilai, aturan diterapkan ke semua prinsipal dengan kunci Tag yang ditentukan.

   b. Pilih Tambahkan pengecualian baru untuk menambahkan pasangan nilai kunci tambahan untuk dikecualikan.

   c. Untuk menghapus pasangan kunci-nilai untuk dikecualikan, pilih Hapus di sebelah pasangan kunci-nilai.

   d. Pilih Simpan perubahan.

6. Pada tab Aturan arsip, Anda dapat membuat, mengedit, atau menghapus aturan arsip untuk penganalisis. Untuk informasi selengkapnya, lihat Aturan arsip.

7. Pada tab Tag, Anda dapat mengelola dan membuat tag untuk penganalisis. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.

Hapus penganalisis akses yang tidak digunakan

Gunakan prosedur berikut untuk menghapus penganalisis akses yang tidak digunakan. Saat Anda menghapus penganalisis, sumber daya tidak lagi dipantau dan tidak ada temuan baru yang dihasilkan. Semua temuan yang dihasilkan oleh penganalisis dihapus.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Access analyzer, pilih Akses yang tidak digunakan.
3. Di bawah Access analyzer, pilih Pengaturan Analyzer.
4. Di bagian Analyzers, pilih nama penganalisis akses yang tidak terpakai untuk dihapus.
5. Pilih Hapus penganalisis.
6. Masukkan hapus dan pilih Hapus untuk mengonfirmasi penghapusan penganalisis.

Melihat dasbor temuan Penganalisis Akses IAM

AWS Identity and Access Management Access Analyzer mengatur temuan akses eksternal, internal, dan tidak terpakai ke dalam dasbor ringkasan visual. Dasbor membantu Anda mendapatkan visibilitas ke dalam penggunaan izin yang efektif dalam skala besar dan mengidentifikasi akun dan AWS sumber daya yang perlu diperhatikan. Anda dapat menggunakan dasbor untuk meninjau temuan berdasarkan AWS organisasi, akun, dan jenis temuan.

Untuk temuan akses eksternal dan internal:

- Dasbor menyoroti pemisahan antara temuan akses publik, temuan akses eksternal, dan temuan akses internal.
- Dasbor menyediakan rincian temuan berdasarkan jenis sumber daya.

Untuk temuan akses yang tidak digunakan:

- Dasbor menyoroti Akun AWS dengan temuan akses yang paling tidak digunakan.
- Dasbor memberikan rincian temuan berdasarkan jenis.

Setelah Anda membuat semua jenis penganalisis akses, IAM Access Analyzer secara otomatis menambahkan temuan baru ke dasbor yang relevan. Ini memungkinkan Anda untuk mengidentifikasi dan memprioritaskan area dengan masalah keamanan paling banyak.

Dasbor ringkasan memberi Anda tampilan tingkat tinggi tentang masalah akses yang terdeteksi oleh IAM Access Analyzer di seluruh lingkungan Anda. AWS Anda kemudian dapat menelusuri temuan individu untuk menyelidiki lebih lanjut dan mengambil tindakan yang tepat untuk menyelesaikannya.

Melihat dasbor ringkasan untuk penganalisis akses eksternal dan internal

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, dasbor ringkasan dapat memakan waktu untuk mencerminkan pembaruan temuan.

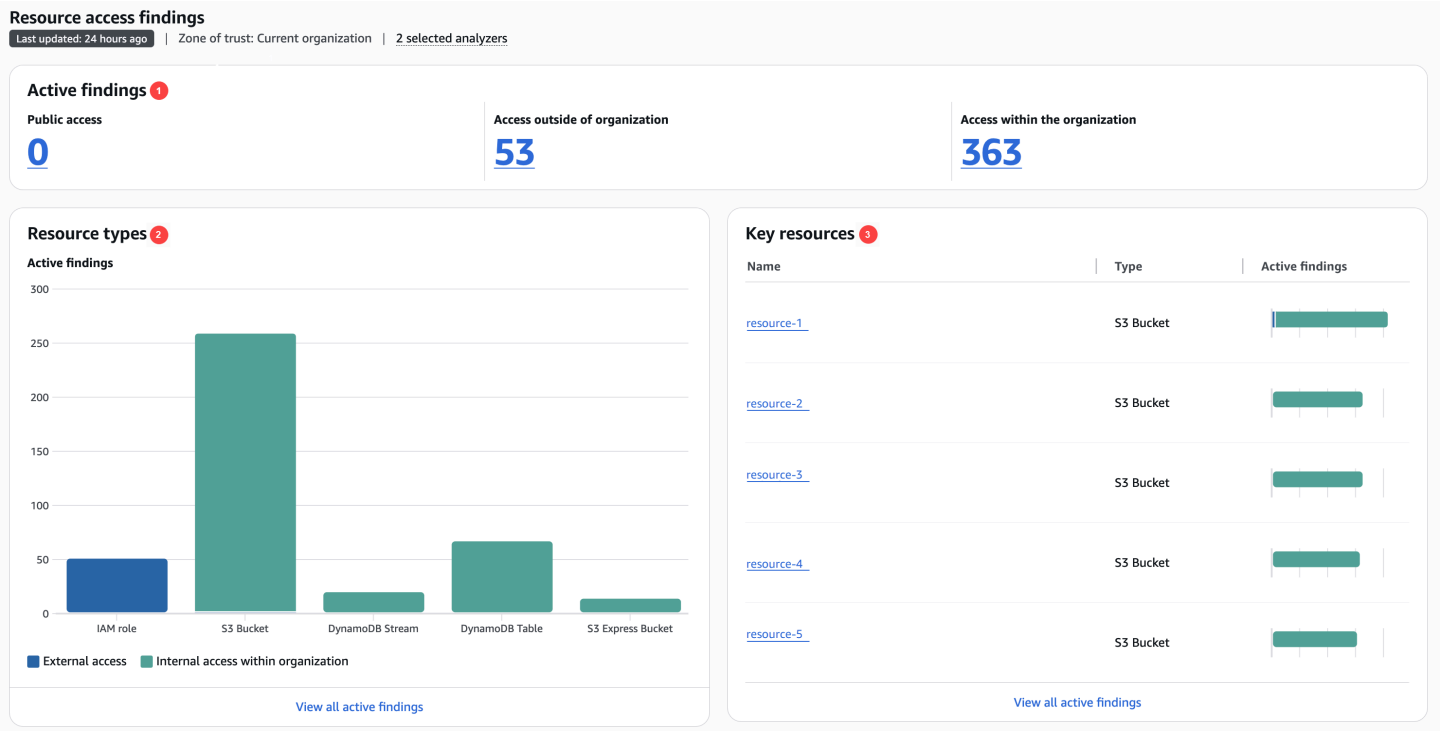
1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Access Analyzer. Jendela Ringkasan ditampilkan.
3. Pilih Pilih penganalisis.
4. Di jendela Select Analyzers, pilih Organisasi atau Akun untuk Zona Kepercayaan.

   > **Note**
   >
   > Hanya akun AWS Organizations manajemen atau administrator yang didelegasikan yang dapat memilih Organisasi sebagai zona kepercayaan.

5. Pilih penganalisis akses eksternal dan internal dari dropdown Resource Access analyzers.

   > **Note**
   >
   > Anda dapat memilih maksimum satu penganalisis akses eksternal dan maksimum satu penganalisis akses internal.

6. Pilih Perbarui. Ringkasan temuan untuk penganalisis akses eksternal dan internal yang dipilih ditampilkan di bagian Temuan akses sumber daya.



Pada gambar sebelumnya, dasbor temuan sumber daya terlihat dari dalam halaman Ringkasan.

1. Bagian temuan aktif mencakup jumlah temuan aktif untuk akses publik, jumlah temuan aktif yang menyediakan akses di luar akun atau organisasi, dan jumlah temuan akses internal aktif untuk penganalisis yang dipilih. Pilih nomor untuk mencantumkan semua temuan aktif dari setiap jenis.
2. Bagian Jenis sumber daya mencakup rincian jenis sumber daya dengan temuan aktif untuk penganalisis yang dipilih. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk analisa yang dipilih.
3. Bagian Sumber daya utama mencakup ringkasan sumber daya utama dengan temuan aktif. Informasi ini membantu Anda memprioritaskan temuan untuk sumber daya penting bisnis Anda. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk analisa yang dipilih.

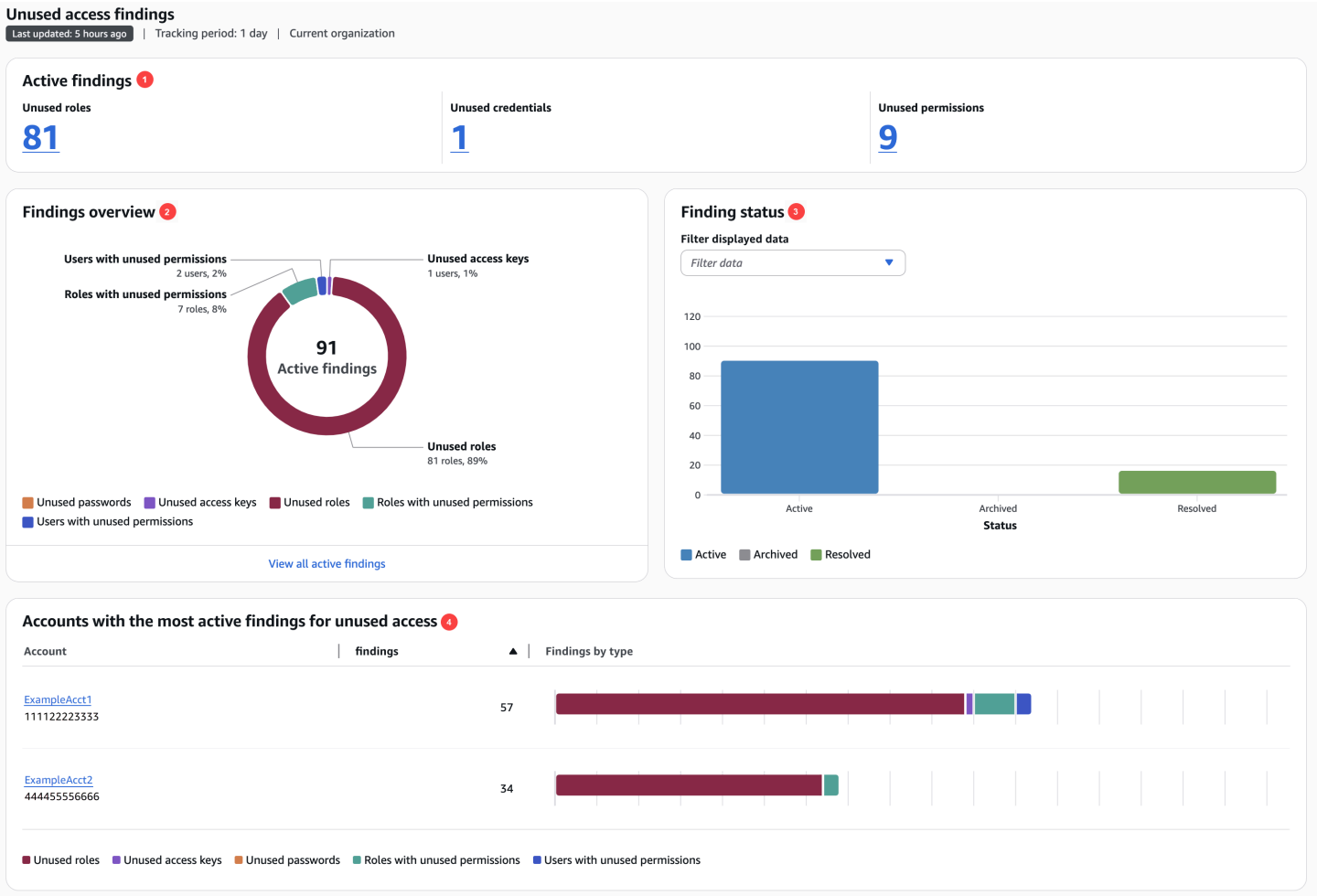
Melihat dasbor ringkasan untuk penganalisis akses yang tidak digunakan

> **Note**
>
> Setelah Anda membuat atau memperbarui penganalisis, berdasarkan jumlah pengguna dan peran, dasbor ringkasan dapat memakan waktu untuk mencerminkan pembaruan temuan.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Access Analyzer. Jendela Access Analyzer Summary ditampilkan.
3. Pilih Pilih penganalisis.
4. Di jendela Select Analyzers, pilih Organisasi atau Akun untuk Zona Kepercayaan.

> **Note**
>
> Hanya akun AWS Organizations manajemen atau administrator yang didelegasikan yang dapat memilih Organisasi sebagai zona kepercayaan.

5. Pilih penganalisis akses yang tidak terpakai dari dropdown penganalisis akses yang tidak digunakan.
6. Pilih Perbarui ringkasan. Ringkasan temuan untuk penganalisis akses tidak terpakai yang dipilih ditampilkan di bagian Temuan akses yang tidak digunakan.



Pada gambar sebelumnya, dasbor temuan akses yang tidak digunakan terlihat dari dalam halaman Ringkasan.

1. Bagian temuan aktif mencakup jumlah temuan aktif untuk peran yang tidak digunakan, kredensi yang tidak digunakan, dan izin yang tidak digunakan di akun atau organisasi Anda. Kredensi yang tidak digunakan mencakup kunci akses yang tidak digunakan dan temuan kata sandi yang tidak digunakan. Izin yang tidak digunakan mencakup pengguna dan peran dengan izin yang tidak digunakan. Pilih nomor untuk mencantumkan semua temuan aktif dari setiap jenis.
2. Bagian ikhtisar Temuan mencakup rincian jenis temuan aktif. Pilih Lihat semua temuan aktif untuk daftar lengkap temuan aktif untuk akun atau organisasi penganalisis.
3. Bagian Status pencarian mencakup rincian status temuan (Aktif, Diarsipkan, dan Terselesaikan) untuk akun atau organisasi Anda. Anda dapat memilih status temuan untuk ditampilkan di Filter data yang ditampilkan tarik-turun.
4. Bagian Akun dengan temuan terbanyak untuk akses yang tidak digunakan hanya ditampilkan jika akun yang dipilih dari penganalisis akses Anda yang tidak digunakan berada di tingkat organisasi. Ini mencakup rincian akun di organisasi Anda dengan temuan paling aktif. Ini bukan daftar lengkap dari setiap akun di organisasi Anda. Analyzer Anda mungkin memiliki temuan aktif untuk akun lain yang tidak tercantum di bagian ini.

Tinjau temuan IAM Access Analyzer

Setelah Anda mengaktifkan IAM Access Analyzer, langkah selanjutnya adalah meninjau temuan apa pun untuk menentukan apakah akses yang diidentifikasi dalam temuan itu disengaja atau tidak disengaja. Anda juga dapat meninjau temuan untuk menentukan temuan serupa untuk akses yang dimaksudkan, dan kemudian membuat aturan arsip untuk mengarsipkan temuan tersebut secara otomatis. Anda juga dapat meninjau temuan yang diarsipkan dan diselesaikan.

Anda harus meninjau semua temuan di akun Anda untuk menentukan apakah akses eksternal, internal, atau tidak terpakai diharapkan dan disetujui. Jika akses yang diidentifikasi dalam temuan diharapkan, Anda dapat mengarsipkan temuan tersebut. Saat Anda mengarsipkan temuan, statusnya diubah menjadi Diarsipkan, dan temuan tersebut dihapus dari daftar temuan aktif. Temuan tidak dihapus. Anda dapat melihat temuan yang diarsipkan kapan saja. Pelajari semua temuan di akun Anda hingga Anda tidak memiliki temuan aktif. Setelah Anda mencapai nol temuan, Anda tahu bahwa setiap temuan Aktif baru yang dihasilkan berasal dari perubahan baru-baru ini di lingkungan Anda.

> **Tip**
>
> Jika Anda memiliki sejumlah besar temuan untuk ditinjau, gunakan perintah IAM Access Analyzer API atau AWS CLI untuk mengambil dan mengelola temuan Anda secara terprogram.

Untuk meninjau temuan aktif untuk semua jenis analisis akses

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Penganalisis akses. Dasbor temuan ditampilkan.
3. Pilih Pilih penganalisis.
4. Di jendela Select analyzers, pilih maksimal satu penganalisis akses eksternal dan maksimal satu penganalisis akses internal dari dropdown Resource Access Analyzers. Pilih penganalisis akses yang tidak terpakai dari dropdown penganalisis akses yang tidak digunakan.
5. Pilih Perbarui ringkasan. Ringkasan temuan aktif untuk penganalisis akses yang dipilih ditampilkan di dasbor. Pilih jenis temuan di bagian Temuan akses sumber daya atau Temuan akses yang tidak digunakan untuk melihat semua temuan aktif dari jenis yang dipilih.

   Untuk informasi lebih lanjut tentang melihat dasbor temuan, lihatMelihat dasbor temuan Penganalisis Akses IAM.

> **Note**
>
> Temuan ditampilkan hanya jika Anda memiliki izin untuk melihat temuan untuk penganalisis.

Temuan akses eksternal dan internal

> **Note**
>
> IAM Access Analyzer mengenakan biaya untuk analisis akses internal berdasarkan jumlah sumber daya yang dipantau per Wilayah per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

1. Di bawah Access Analyzer, pilih Analisis sumber daya.

2. Pilih Pilih penganalisis.
3. Di jendela Select analyzers, pilih maksimal satu penganalisis akses eksternal dan maksimal satu penganalisis akses internal dari dropdown Resource Access Analyzers.
4. Pilih Perbarui ringkasan.

   Halaman Analisis sumber daya menampilkan detail berikut tentang sumber daya dengan temuan aktif untuk penganalisis akses yang dipilih:

Nama

Nama sumber daya dengan temuan aktif.

Jenis

Jenis sumber daya.

Akun pemilik

Kolom ini hanya ditampilkan jika Anda menggunakan organisasi sebagai zona kepercayaan untuk satu atau lebih penganalisis yang dipilih. Akun dalam organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.

Temuan aktif

Representasi visual dari jumlah dan jenis temuan aktif untuk sumber daya. Arahkan kursor ke lapangan untuk menampilkan informasi lebih lanjut tentang temuan untuk sumber daya.

Akses publik

Menunjukkan apakah salah satu temuan untuk sumber daya memungkinkan akses publik.

Temuan akses yang tidak digunakan

> **Note**
>
> IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

1. Di bawah Access Analyzer, pilih Akses yang tidak digunakan.

2. Pilih Pilih penganalisis.
3. Di jendela Select analyzers, pilih penganalisis akses yang tidak digunakan dari dropdown Penganalisis akses yang tidak digunakan.
4. Pilih Perbarui ringkasan.

   Halaman akses yang tidak digunakan menampilkan detail berikut tentang entitas IAM yang menghasilkan temuan untuk penganalisis akses yang dipilih:

Menemukan ID

ID unik yang ditetapkan untuk temuan. Pilih ID temuan untuk menampilkan detail tambahan tentang entitas IAM yang menghasilkan temuan.

Menemukan jenis

Jenis pencarian akses yang tidak digunakan: Kunci akses yang tidak digunakan, Kata sandi yang tidak digunakan, Izin yang tidak digunakan, atau Peran yang tidak digunakan.

Entitas IAM

Entitas IAM dilaporkan dalam temuan tersebut. Ini bisa menjadi pengguna atau peran IAM.

Akun AWS ID

Kolom ini hanya ditampilkan jika Anda mengatur penganalisis untuk semua yang Akun AWS ada di organisasi. Akun AWS Dalam organisasi yang memiliki entitas IAM dilaporkan dalam temuan.

Terakhir diperbarui

Terakhir kali entitas IAM dilaporkan dalam temuan diperbarui, atau ketika entitas dibuat jika tidak ada pembaruan yang dilakukan.

Status

Status temuan (Aktif, Diarsipkan, atau Diselesaikan).

Filter temuan IAM Access Analyzer

Pemfilteran default untuk halaman temuan adalah untuk menampilkan semua temuan aktif. Untuk melihat semua temuan, pilih Semua dari tarik-turun Status. Untuk melihat temuan yang diarsipkan, pilih Diarsipkan. Untuk melihat temuan yang diselesaikan, pilih Terselesaikan. Saat pertama kali mulai menggunakan IAM Access Analyzer, tidak ada temuan yang diarsipkan.

Gunakan filter untuk menampilkan hanya temuan yang memenuhi kriteria properti yang ditentukan. Untuk membuat filter, pilih properti yang akan difilter, lalu pilih apakah properti sama atau berisi nilai, lalu masukkan atau pilih nilai properti untuk difilter.

Untuk daftar kunci filter yang dapat Anda gunakan untuk membuat atau memperbarui aturan arsip, lihat Kunci filter IAM Access Analyzer.

Memfilter sumber daya dengan temuan aktif

Anda dapat melihat dan memfilter temuan aktif berdasarkan sumber daya untuk maksimum satu penganalisis akses eksternal dan maksimum satu penganalisis akses internal.

Untuk menyaring sumber daya dengan temuan aktif

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Analisis sumber daya.
3. Untuk memfilter berdasarkan nama sumber daya, ketik semua atau sebagian nama sumber daya di kotak pencarian.
4. Di dropdown tipe akses Filter, pilih jenis akses:
   - Semua jenis — menampilkan sumber daya dengan semua jenis temuan akses.
   - Akses publik — hanya menampilkan sumber daya dengan temuan akses publik.
   - Akses eksternal — hanya menampilkan sumber daya dengan temuan akses eksternal.
   - Akses internal dalam organisasi — hanya menampilkan sumber daya dengan temuan akses internal.
5. Di menu tarik-turun Jenis sumber daya filter, pilih jenis sumber daya untuk hanya menampilkan sumber daya dari jenis yang dipilih.

Memfilter temuan akses eksternal

Untuk menyaring temuan akses eksternal

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih pengaturan Analyzer dan kemudian pilih penganalisis akses eksternal di bagian Analyzers.
3. Pilih Lihat temuan.
4. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
5. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.

6. Pilih nilai yang sesuai dengan properti. Hanya temuan dengan nilai tersebut dalam temuan yang ditampilkan.

   Misalnya, pilih Resource sebagai properti, lalu pilih Resource:, lalu ketik sebagian atau seluruh nama bucket, lalu tekan Enter. Hanya temuan untuk bucket yang cocok dengan kriteria filter yang ditampilkan. Untuk membuat filter yang hanya menampilkan temuan untuk sumber daya yang memungkinkan akses publik, Anda dapat memilih properti Akses publik, lalu pilih Akses publik =, lalu pilih Akses publik = true.

Anda dapat menambahkan properti tambahan untuk memfilter temuan yang ditampilkan lebih lanjut. Saat Anda menambahkan properti tambahan, hanya temuan yang cocok dengan semua ketentuan dalam filter yang ditampilkan. Menetapkan filter untuk menampilkan temuan yang cocok dengan satu properti ATAU properti lain tidak didukung. Pilih Hapus filter untuk menghapus filter apa pun yang telah Anda tetapkan dan menampilkan semua temuan dengan status yang ditentukan untuk penganalisis Anda.

Beberapa bidang hanya ditampilkan saat Anda melihat temuan untuk penganalisis dengan organisasi sebagai zona kepercayaannya.

Properti berikut tersedia untuk mendefinisikan filter untuk akses eksternal:

- Akses publik – Untuk memfilter berdasarkan temuan untuk sumber daya yang memungkinkan akses publik, filter berdasarkan Akses publik, lalu pilih Akses publik: benar.
- Sumber Daya – Untuk memfilter berdasarkan sumber daya, ketik semua atau sebagian nama sumber daya.
- Jenis Sumber Daya – Untuk memfilter berdasarkan jenis sumber daya, pilih jenis dari daftar yang ditampilkan.
- Akun Pemilik Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan akun di organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.
- Pembatasan Kebijakan Kontrol Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan jenis pembatasan yang diterapkan oleh kebijakan kontrol sumber daya Organizations (RCP). Untuk mempelajari selengkapnya, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
  - Gagal mengevaluasi RCP: Ada kesalahan saat mengevaluasi RCP.
  - Tidak berlaku: Tidak ada RCP yang membatasi sumber daya atau prinsipal ini. Ini juga termasuk sumber daya RCPs yang belum didukung.

  - Berlaku: Administrator organisasi Anda telah menetapkan batasan melalui RCP yang memengaruhi sumber daya atau jenis sumber daya. Hubungi administrator organisasi Anda untuk detail selengkapnya.
- AWS Akun — Gunakan properti ini untuk memfilter dengan akses Akun AWS yang diberikan di bagian Utama dari pernyataan kebijakan. Untuk memfilter berdasarkan Akun AWS, ketik semua atau sebagian dari Akun AWS ID 12 digit, atau semua atau sebagian dari ARN akun penuh dari pengguna AWS eksternal atau peran yang memiliki akses ke sumber daya di akun saat ini.
- Pengguna Canonical — Untuk memfilter menurut pengguna kanonik, ketikkan ID pengguna kanonik seperti yang ditentukan untuk bucket Amazon S3. Untuk mempelajari selengkapnya, lihat Pengidentifikasi Akun AWS.
- Pengguna yang Digabungkan – Untuk memfilter berdasarkan pengguna yang digabungkan, ketik semua atau sebagian ARN dari identitas yang digabungkan. Untuk mempelajari, lihat Penyedia dan Gabungan Identitas.
- Menemukan ID — Untuk memfilter dengan menemukan ID, ketik semua atau sebagian dari ID pencarian.
- Kesalahan — Untuk memfilter berdasarkan jenis kesalahan, pilih Akses Ditolak atau Kesalahan Internal.
- ARN Utama — Gunakan properti ini untuk memfilter ARN prinsipal (pengguna, peran, atau grup IAM) yang digunakan dalam aws: condition key. PrincipalArn Untuk memfilter menurut Principal ARN, ketik semua atau sebagian ARN pengguna, peran, atau grup IAM dari eksternal Akun AWS yang dilaporkan dalam sebuah temuan.
- Principal OrgID — Untuk memfilter berdasarkan Principal OrGid, ketik semua atau sebagian ID organisasi yang terkait dengan prinsipal eksternal yang termasuk dalam AWS organisasi yang ditentukan sebagai kondisi dalam temuan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Principal OrgPaths — Untuk memfilter menurut Principal OrgPaths, ketikkan semua atau sebagian ID untuk AWS organisasi atau unit organisasi (OU) yang memungkinkan akses ke semua kepala sekolah eksternal yang merupakan anggota akun dari organisasi tertentu atau OU sebagai syarat dalam kebijakan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Akun Sumber — Untuk memfilter pada Akun Sumber, ketik semua atau sebagian Akun AWS ID yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan di. AWS Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.

- ARN Sumber – Untuk memfilter berdasarkan ARN Sumber, ketikkan semua atau sebagian ARN yang ditentukan sebagai ketentuan di temuan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- IP Sumber – Untuk memfilter berdasarkan IP Sumber, ketikkan semua atau sebagian alamat IP yang memungkinkan akses entitas eksternal ke sumber daya dalam akun saat menggunakan alamat IP yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber OrgID — Untuk memfilter berdasarkan Sumber OrgID, ketik semua atau sebagian ID organisasi yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan di. AWS Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber OrgPaths — Untuk memfilter menurut Sumber OrgPaths, ketik semua atau sebagian unit organisasi (OU) yang terkait dengan sumber daya, seperti yang digunakan dalam beberapa izin lintas layanan di AWS. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- VPC Sumber – Untuk memfilter berdasarkan VPC Sumber, ketikkan semua atau sebagian ID VPC yang memungkinkan entitas eksternal mengakses sumber daya dalam akun saat ini saat menggunakan VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber VPC ARN — Untuk memfilter berdasarkan Sumber VPC ARN, ketik semua atau sebagian dari VPC ARN yang memungkinkan entitas eksternal mengakses sumber daya di akun saat menggunakan VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Sumber VPCE - Untuk memfilter berdasarkan Sumber VPCE, ketik semua atau sebagian ID titik akhir VPC yang memungkinkan entitas eksternal mengakses sumber daya di akun saat menggunakan titik akhir VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- Akun VPCE — Untuk memfilter berdasarkan Akun VPCE, ketik semua atau sebagian ID 12 digit Akun AWS yang memiliki entitas eksternal titik akhir VPC dan memungkinkan entitas eksternal mengakses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- VPCE OrgID — Untuk memfilter berdasarkan VPCE OrgID, ketik semua atau sebagian ID organisasi yang memiliki entitas eksternal titik akhir VPC dan memungkinkan entitas eksternal mengakses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- VPCE OrgPaths — Untuk memfilter berdasarkan VPCE OrgPaths, ketik semua atau sebagian unit organisasi (OU) yang memiliki entitas eksternal titik akhir VPC dan memungkinkan entitas eksternal mengakses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.

- User ID — Untuk memfilter berdasarkan User ID, ketik semua atau sebagian ID pengguna pengguna IAM dari eksternal Akun AWS yang diizinkan mengakses sumber daya di akun saat ini. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.
- ID Kunci KMS - Untuk memfilter berdasarkan ID kunci KMS, ketik semua atau sebagian ID kunci untuk kunci KMS yang ditentukan sebagai kondisi untuk akses objek Amazon S3 yang AWS KMS dienkripsi di akun Anda saat ini.
- Mode Sesi — Untuk memfilter berdasarkan mode sesi untuk bucket direktori Amazon S3 (`ReadOnly`atau`ReadWrite`, ketik semua atau sebagian dari mode sesi. Untuk mempelajari selengkapnya, lihat CreateSessiondi Referensi API Amazon Simple Storage Service.
- Google Audience – Untuk memfilter oleh Google Audience, ketikkan semua atau sebagian ID aplikasi Google yang ditentukan sebagai ketentuan untuk akses peran IAM di akun Anda saat ini. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.
- Pemirsa Cognito — Untuk memfilter menurut audiens Amazon Cognito, ketik semua atau sebagian ID kumpulan identitas Amazon Cognito yang ditentukan sebagai syarat untuk akses peran IAM di akun Anda saat ini. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.
- Akun Penelepon — Akun AWS ID akun yang memiliki atau berisi entitas pemanggil, seperti peran IAM, pengguna, atau pengguna root akun. Ini digunakan oleh panggilan layanan AWS KMS. Untuk memfilter berdasarkan akun penelepon, ketik semua atau sebagian Akun AWS ID.
- ID Aplikasi Facebook – Untuk memfilter berdasarkan ID Aplikasi Facebook, ketikkan semua atau sebagian dari ID aplikasi Facebook (atau ID situs) yang ditentukan sebagai ketentuan untuk memungkinkan Login dengan akses gabungan Facebook ke peran IAM di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat bagian id dalam kunci konteks IAM dan AWS STS kondisi.
- ID Aplikasi Amazon – Untuk memfilter dengan ID Aplikasi Amazon, ketikkan semua atau sebagian dari ID aplikasi Amazon (atau ID situs) yang ditentukan sebagai ketentuan untuk memungkinkan akses federasi Login with Amazon ke peran IAM di akun Anda saat ini. Untuk mempelajari lebih lanjut, lihat bagian id dalam kunci konteks IAM dan AWS STS kondisi.
- Token Sumber Peristiwa Lambda – Untuk memfilter Token Sumber Peristiwa Lambda yang diteruskan dengan integrasi Alexa, ketikkan semua atau sebagian string token.

Menyaring temuan akses internal

Untuk menyaring temuan akses internal

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Pilih pengaturan Analyzer dan kemudian pilih penganalisis akses internal di bagian Analyzers.
3. Pilih Lihat temuan.
4. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
5. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
6. Pilih nilai yang sesuai dengan properti. Hanya temuan dengan nilai tersebut dalam temuan yang ditampilkan.

   Misalnya, pilih Resource sebagai properti, lalu pilih Resource:, lalu ketik sebagian atau seluruh nama bucket, lalu tekan Enter. Hanya temuan untuk bucket yang cocok dengan kriteria filter yang ditampilkan.

Anda dapat menambahkan properti tambahan untuk memfilter temuan yang ditampilkan lebih lanjut. Saat Anda menambahkan properti tambahan, hanya temuan yang cocok dengan semua ketentuan dalam filter yang ditampilkan. Menetapkan filter untuk menampilkan temuan yang cocok dengan satu properti ATAU properti lain tidak didukung. Pilih Hapus filter untuk menghapus filter apa pun yang telah Anda tetapkan dan menampilkan semua temuan dengan status yang ditentukan untuk penganalisis Anda.

Beberapa bidang hanya ditampilkan saat Anda melihat temuan untuk penganalisis dengan organisasi sebagai zona kepercayaannya.

Bidang berikut hanya ditampilkan saat Anda melihat temuan untuk penganalisis yang memantau akses internal:

- Sumber Daya – Untuk memfilter berdasarkan sumber daya, ketik semua atau sebagian nama sumber daya.
- Jenis Sumber Daya – Untuk memfilter berdasarkan jenis sumber daya, pilih jenis dari daftar yang ditampilkan.
- Akun Pemilik Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan akun di organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.
- Mencari id — Untuk memfilter dengan menemukan ID, ketik semua atau sebagian dari ID pencarian.

Memfilter temuan akses yang tidak digunakan

Untuk menyaring temuan akses yang tidak digunakan

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak digunakan dan kemudian pilih penganalisis di menu tarik-turun View analyzer.
3. Pilih kotak pencarian untuk menampilkan daftar properti yang tersedia.
4. Pilih properti yang akan digunakan untuk memfilter temuan yang ditampilkan.
5. Pilih nilai yang sesuai dengan properti. Hanya temuan dengan nilai tersebut dalam temuan yang ditampilkan.

   Misalnya, pilih Jenis temuan sebagai properti, lalu pilih Jenis temuan =, lalu pilih Jenis temuan = Peran yang tidak digunakan. Hanya temuan dengan jenis peran yang tidak digunakan yang ditampilkan.

Anda dapat menambahkan properti tambahan untuk memfilter temuan yang ditampilkan lebih lanjut. Saat Anda menambahkan properti tambahan, hanya temuan yang cocok dengan semua ketentuan dalam filter yang ditampilkan. Menetapkan filter untuk menampilkan temuan yang cocok dengan satu properti ATAU properti lain tidak didukung. Pilih Hapus filter untuk menghapus filter apa pun yang telah Anda tetapkan dan menampilkan semua temuan dengan status yang ditentukan untuk penganalisis Anda.

Bidang berikut hanya ditampilkan saat Anda melihat temuan untuk penganalisis yang memantau akses yang tidak digunakan:

- Jenis temuan — Untuk memfilter berdasarkan jenis pencarian, filter berdasarkan jenis Temuan dan kemudian pilih jenis temuan.
- Sumber Daya – Untuk memfilter berdasarkan sumber daya, ketik semua atau sebagian nama sumber daya.
- Jenis Sumber Daya – Untuk memfilter berdasarkan jenis sumber daya, pilih jenis dari daftar yang ditampilkan.
- Akun Pemilik Sumber Daya — Gunakan properti ini untuk memfilter berdasarkan akun di organisasi yang memiliki sumber daya yang dilaporkan dalam temuan.
- Mencari id — Untuk memfilter dengan menemukan ID, ketik semua atau sebagian dari ID pencarian.

Arsipkan temuan IAM Access Analyzer

Ketika Anda mendapatkan temuan untuk akses ke sumber daya yang disengaja, Anda dapat mengarsipkan temuan. Misalnya, pencarian akses eksternal atau internal untuk bucket Amazon S3 yang diakses untuk alur kerja yang disetujui atau pencarian akses yang tidak digunakan untuk kunci akses yang mungkin masih diperlukan. Ketika Anda mengarsipkan temuan, itu dihapus dari daftar temuan aktif. Temuan yang diarsipkan tidak dihapus. Anda dapat memfilter halaman Temuan untuk menampilkan temuan yang diarsipkan, dan membatalkan arsipkannya kapan saja.

Untuk mengarsipkan temuan dari laman Temuan

1. Pilih kotak centang di samping satu atau beberapa temuan untuk diarsipkan.
2. Pilih Tindakan dan kemudian pilih Arsip.

   Konfirmasi ditampilkan di bagian atas layar.

Untuk mengarsipkan temuan dari halaman Detail Temuan

1. Pilih ID Temuan untuk mengarsip temuan.

2. Pilih Arsipkan.

   Konfirmasi ditampilkan di bagian atas layar.

Untuk menghapus arsip temuan, ulangi langkah sebelumnya, tetapi pilih Batalkan Arsip alih-alih Arsipkan. Saat Anda menghapus arsip temuan, status diatur ke Aktif.

Selesaikan temuan IAM Access Analyzer

Menyelesaikan temuan sumber daya

Untuk menyelesaikan temuan akses eksternal dan internal yang dihasilkan dari akses yang tidak diinginkan, Anda harus mengubah pernyataan kebijakan untuk menghapus izin yang memungkinkan akses ke sumber daya yang diidentifikasi.

Untuk temuan yang terkait dengan bucket Amazon S3, gunakan konsol Amazon S3 untuk mengonfigurasi izin di bucket.

Untuk IAM role, gunakan konsol IAM untuk mengubah kebijakan kepercayaan untuk IAM role yang terdaftar.

Untuk sumber daya lain yang didukung, gunakan konsol untuk mengubah pernyataan kebijakan yang menghasilkan temuan yang dihasilkan.

Setelah membuat perubahan untuk menyelesaikan temuan sumber daya, seperti memodifikasi kebijakan yang diterapkan ke peran IAM, IAM Access Analyzer akan memindai sumber daya lagi. Jika akses ke sumber daya dihapus, status temuan diubah menjadi Terselesaikan. Temuan ini kemudian akan ditampilkan dalam daftar temuan yang diselesaikan alih-alih daftar temuan aktif.

> **Note**
>
> Ini tidak berlaku untuk temuan Kesalahan. Ketika IAM Access Analyzer tidak dapat menganalisis sumber daya, itu akan menghasilkan temuan kesalahan. Jika Anda menyelesaikan masalah yang mencegah IAM Access Analyzer menganalisis sumber daya, temuan kesalahan akan dihapus sepenuhnya alih-alih mengubah ke temuan yang diselesaikan. Untuk informasi selengkapnya, lihat Temuan kesalahan IAM Access Analyzer.

Jika perubahan yang Anda buat menghasilkan akses eksternal atau internal ke sumber daya, tetapi dengan cara yang berbeda, seperti dengan prinsipal yang berbeda atau untuk izin yang berbeda, IAM Access Analyzer akan menyelesaikan temuan asli dan menghasilkan temuan Aktif baru. Jika perubahan yang Anda buat mengakibatkan kesalahan internal atau kesalahan akses ditolak, semua temuan non-kesalahan aktif yang terkait dengan akses spesifik sumber daya diselesaikan dan temuan kesalahan baru dihasilkan.

> **Note**
>
> Untuk penganalisis akses eksternal, mungkin diperlukan waktu hingga 30 menit setelah kebijakan dimodifikasi untuk IAM Access Analyzer untuk menganalisis sumber daya lagi dan kemudian memperbarui temuannya.
> Untuk penganalisis akses internal, mungkin diperlukan beberapa menit atau jam untuk IAM Access Analyzer untuk menganalisis sumber daya lagi dan kemudian memperbarui temuan. IAM Access Analyzer secara otomatis memindai ulang semua kebijakan setiap 24 jam.
> Temuan yang diselesaikan dihapus 90 hari setelah pembaruan terakhir ke status temuan.

Menyelesaikan temuan akses yang tidak terpakai

IAM Access Analyzer menyediakan langkah-langkah yang direkomendasikan untuk menyelesaikan temuan penganalisis akses yang tidak digunakan berdasarkan jenis temuan.

Setelah Anda membuat perubahan untuk menyelesaikan temuan akses yang tidak digunakan, status temuan diubah menjadi Terselesaikan saat berikutnya penganalisis akses yang tidak digunakan berjalan. Temuan ini tidak lagi ditampilkan dalam daftar temuan aktif dan sebagai gantinya ditampilkan dalam daftar temuan yang diselesaikan. Jika Anda membuat perubahan yang hanya sebagian membahas temuan akses yang tidak digunakan, temuan yang ada diubah menjadi Terselesaikan tetapi temuan baru dihasilkan. Misalnya, jika Anda menghapus hanya beberapa izin yang tidak digunakan dalam temuan, tetapi tidak semuanya.

IAM Access Analyzer mengenakan biaya untuk analisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per bulan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Menyelesaikan temuan izin yang tidak digunakan

Untuk temuan izin yang tidak digunakan, IAM Access Analyzer dapat merekomendasikan kebijakan untuk dihapus dari pengguna atau peran IAM dan memberikan kebijakan baru untuk mengganti kebijakan izin yang ada. Rekomendasi kebijakan tidak didukung untuk skenario berikut:

- Temuan izin yang tidak digunakan adalah untuk pengguna IAM yang berada dalam grup pengguna.
- Temuan izin yang tidak digunakan adalah untuk peran IAM untuk IAM Identity Center.
- Temuan izin yang tidak digunakan memiliki kebijakan izin yang sudah ada yang menyertakan elemen. `notAction`

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused Permissions.
4. Di bagian Rekomendasi, jika ada kebijakan yang tercantum di kolom Kebijakan yang disarankan, pilih Kebijakan pratinjau untuk melihat kebijakan yang ada dengan kebijakan yang disarankan untuk mengganti kebijakan yang ada. Jika ada beberapa kebijakan yang direkomendasikan, Anda dapat memilih Kebijakan Berikutnya dan Kebijakan Sebelumnya untuk melihat setiap kebijakan yang ada dan yang direkomendasikan.

5. Pilih Unduh JSON untuk mengunduh file.zip dengan file JSON dari semua kebijakan yang disarankan.
6. Buat dan lampirkan kebijakan yang direkomendasikan ke pengguna atau peran IAM. Untuk informasi selengkapnya, lihat Mengubah izin untuk pengguna (konsol) dan Memodifikasi kebijakan izin peran (konsol).
7. Hapus kebijakan yang tercantum dalam kolom Kebijakan izin yang ada dari pengguna atau peran IAM. Untuk informasi selengkapnya, lihat Menghapus izin dari pengguna (konsol) dan Memodifikasi kebijakan izin peran (konsol).

Menyelesaikan temuan peran yang tidak digunakan

Untuk temuan peran yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menghapus peran IAM yang tidak digunakan.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused role.
4. Di bagian Rekomendasi, tinjau detail peran IAM.
5. Hapus peran IAM. Untuk informasi selengkapnya, lihat Menghapus peran IAM (konsol).

Menyelesaikan temuan kunci akses yang tidak digunakan

Untuk temuan kunci akses yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menonaktifkan atau menghapus kunci akses yang tidak digunakan.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Finding of Unused access keys.
4. Di bagian Rekomendasi, tinjau detail kunci akses.
5. Nonaktifkan atau hapus kunci akses. Untuk informasi selengkapnya, lihat Mengelola kunci akses (konsol).

Menyelesaikan temuan kata sandi yang tidak digunakan

Untuk temuan kata sandi yang tidak digunakan, IAM Access Analyzer merekomendasikan untuk menghapus kata sandi yang tidak digunakan untuk pengguna IAM.

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Akses yang tidak terpakai.
3. Pilih temuan dengan jenis Menemukan kata sandi yang tidak digunakan.
4. Di bagian Rekomendasi, tinjau detail pengguna IAM.
5. Hapus kata sandi untuk pengguna IAM. Untuk informasi selengkapnya, lihat Membuat, mengubah, atau menghapus kata sandi pengguna (konsol) IAM.

Temuan kesalahan IAM Access Analyzer

Ketika IAM Access Analyzer menganalisis sumber daya, biasanya menghasilkan temuan yang menunjukkan siapa yang memiliki akses ke sumber daya Anda. Namun, dalam beberapa kasus, penganalisis mungkin mengalami masalah yang mencegahnya menyelesaikan analisis. Dalam situasi ini, IAM Access Analyzer menghasilkan temuan kesalahan sebagai gantinya.

Temuan kesalahan menunjukkan bahwa IAM Access Analyzer tidak dapat menyelesaikan analisis untuk sumber daya tertentu atau untuk pasangan sumber daya utama tertentu. Temuan ini membantu Anda mengidentifikasi sumber daya yang mungkin perlu diperhatikan untuk memastikan analisis yang tepat.

Temuan kesalahan akses eksternal

Penganalisis akses eksternal, yang mengidentifikasi sumber daya yang dibagikan di luar akun atau organisasi Anda, dapat menghasilkan dua jenis temuan kesalahan:

- INTERNAL_ERROR — Menunjukkan bahwa IAM Access Analyzer mengalami masalah internal saat menganalisis sumber daya. Ini bisa disebabkan oleh keterbatasan layanan atau masalah sementara.

```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
```

```
 ],
 "resource": "arn:aws:iam::941407043048:role/TestAccessAnalyzer",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR",
 "createdAt": "2022-07-14T01:31:43.085000+00:00",
 "resourceType": "AWS::IAM::Role",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.109000+00:00",
 "id": "4b035c7d-b7d2-40e4-a6c3-9887d1a995df",
 "updatedAt": "2022-07-14T01:31:43.085000+00:00"
}
```

- ACCESS_DENIED — Menunjukkan bahwa IAM Access Analyzer tidak memiliki izin yang diperlukan untuk menganalisis sumber daya. Ini biasanya terjadi ketika peran terkait layanan (SLR) untuk IAM Access Analyzer ditolak akses ke sumber daya.

```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:kms:us-west-2:941407043048:key/01cae123-
b7f2-4488-9a05-0070a072ea2c",
 "status": "ACTIVE",
 "error": "ACCESS_DENIED",
 "createdAt": "2022-07-14T01:31:43.104000+00:00",
 "resourceType": "AWS::KMS::Key",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.090000+00:00",
 "id": "7ef6f04a-9d2c-4038-9cc0-2a5f00a4d8f8",
 "updatedAt": "2022-07-14T01:31:43.104000+00:00"
}
```

Temuan kesalahan akses internal

Penganalisis akses internal, yang mengidentifikasi akses dalam akun atau organisasi Anda, dapat menghasilkan empat jenis temuan kesalahan:

- PRINCIPAL_LIMIT_EXCEEDED — Dihasilkan ketika lebih dari 3.000 kepala sekolah memiliki akses ke sumber daya penting. Kesalahan ini membantu Anda mengidentifikasi sumber daya dengan akses yang terlalu luas yang mungkin perlu dibatasi.

  Jika Anda membuat perubahan pada sumber daya atau prinsip di lingkungan Anda yang membawa jumlah prinsipal di bawah batas, penganalisis akan menghasilkan temuan normal selama pemindaian berikutnya, dan temuan kesalahan akan ditandai sebagai diselesaikan.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Kesalahan tingkat sumber daya (INTERNAL_ERROR atau ACCESS_DENIED) — Mirip dengan kesalahan akses eksternal, ini menunjukkan bahwa penganalisis tidak dapat menganalisis sumber daya tertentu karena masalah internal atau masalah izin. Ketika kesalahan tingkat sumber daya terjadi, penganalisis menghasilkan satu temuan kesalahan untuk sumber daya alih-alih temuan normal.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
```

```
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Kesalahan tingkat utama (INTERNAL_ERROR atau ACCESS_DENIED) - Menunjukkan bahwa penganalisis tidak dapat menganalisis akses untuk prinsipal tertentu ke sumber daya tertentu. Tidak seperti kesalahan tingkat sumber daya, sumber daya dapat memiliki temuan normal untuk beberapa prinsip dan temuan kesalahan untuk prinsip lainnya.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {
    "principal": {
     "AWS": "arn:aws:iam::111122223333:role/MyRole_1"
    },
    "principalOwnerAccount": "111122223333",
    "principalType": "IAM_ROLE",
    "accessType": "INTRA_ACCOUNT"
   }
  }
 ]
}
```

- PRINCIPAL_ERRORS_LIMIT_EXCEEDED — Dihasilkan ketika ada terlalu banyak temuan kesalahan tingkat utama untuk satu sumber daya. Ini adalah temuan kesalahan tingkat sumber daya yang mungkin muncul di samping temuan normal untuk sumber daya yang sama.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_ERRORS_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "resourceControlPolicyRestriction": "NOT_APPLICABLE",
 "serviceControlPolicyRestriction": "NOT_APPLICABLE",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```


Menyelesaikan temuan kesalahan

Jika Anda menyelesaikan masalah yang mencegah IAM Access Analyzer menganalisis sumber daya, temuan kesalahan akan dihapus sepenuhnya alih-alih mengubah ke temuan yang diselesaikan.

Untuk mengatasi temuan kesalahan, pertimbangkan pendekatan berikut berdasarkan jenis kesalahan:

- Untuk kesalahan ACCESS_DENIED, verifikasi bahwa peran terkait layanan IAM Access Analyzer memiliki izin yang diperlukan untuk mengakses sumber daya.
- Untuk kesalahan PRINCIPAL_LIMIT_EXCEEDED, tinjau kebijakan akses sumber daya dan pertimbangkan untuk membatasi akses ke lebih sedikit prinsipal.
- Untuk temuan INTERNAL_ERROR, Anda mungkin perlu menunggu siklus analisis berikutnya atau menghubungi AWS dukungan jika masalah berlanjut.
- Untuk PRINCIPAL_ERRORS_LIMIT_EXCEEDED, tinjau dan sederhanakan pola akses untuk sumber daya yang terpengaruh.

Setelah membuat perubahan untuk mengatasi masalah mendasar, IAM Access Analyzer akan mencoba menganalisis sumber daya lagi selama siklus pemindaian berikutnya.

IAM Access Analyzer mendukung jenis sumber daya untuk akses eksternal dan internal

Untuk penganalisis akses eksternal dan internal, IAM Access Analyzer menganalisis kebijakan berbasis sumber daya yang diterapkan ke sumber AWS daya di Wilayah tempat Anda mengaktifkan IAM Access Analyzer. Ini hanya menganalisis kebijakan berbasis sumber daya. Untuk detail tentang cara IAM Access Analyzer menghasilkan temuan untuk setiap jenis sumber daya, tinjau informasi jenis sumber daya.

> **Note**
>
> Jenis sumber daya yang didukung yang tercantum adalah untuk penganalisis akses eksternal dan internal. Penganalisis akses internal tidak mendukung semua jenis sumber daya yang didukung oleh penganalisis akses eksternal. Penganalisis akses yang tidak digunakan hanya mendukung pengguna dan peran IAM. Untuk informasi selengkapnya, lihat Memahami cara kerja temuan IAM Access Analyzer.

Jenis sumber daya yang didukung untuk akses eksternal

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- AWS Identity and Access Management peran
- AWS Key Management Service kunci
- AWS Lambda fungsi dan lapisan
- Antrean Amazon Simple Queue Service
- AWS Secrets Manager rahasia
- Topik Amazon Simple Notification Service
- Cuplikan volume Amazon Elastic Block Store
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon

- Repositori Registri Wadah Elastis Amazon
- Sistem file Amazon Elastic File System
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Jenis sumber daya yang didukung untuk akses internal

- Bucket Amazon Simple Storage Service
- Ember direktori Layanan Penyimpanan Sederhana Amazon
- Cuplikan DB Layanan Amazon Relational Database Service
- Cuplikan cluster DB Layanan Relational Database Service Amazon
- Aliran Amazon DynamoDB
- Tabel Amazon DynamoDB

Bucket Amazon Simple Storage Service

Saat IAM Access Analyzer menganalisis bucket Amazon S3 untuk penganalisis akses eksternal, analisis ini akan menghasilkan temuan saat kebijakan bucket Amazon S3, daftar kontrol akses (ACL), atau titik akses, termasuk titik akses Multi-wilayah, diterapkan ke bucket memberikan akses ke entitas eksternal. Entitas eksternal adalah penanggung jawab atau entitas lain yang dapat Anda gunakan untuk buat filter yang tidak berada dalam zona kepercayaan Anda. Misalnya, jika kebijakan bucket memberikan akses ke akun lain atau mengizinkan akses publik, IAM Access Analyzer akan menghasilkan temuan. Namun, jika Anda mengaktifkan Blokir Akses Publik di bucket, Anda dapat memblokir akses di tingkat akun atau tingkat bucket.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan saat prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke bucket Amazon S3 yang ditentukan.

> **Note**
>
> IAM Access Analyzer tidak menganalisis kebijakan titik akses yang dilampirkan ke jalur akses lintas akun karena titik akses dan kebijakannya berada di luar akun penganalisis. IAM Access Analyzer menghasilkan temuan publik saat bucket mendelegasikan akses ke titik akses lintas akun dan Blokir Akses Publik tidak diaktifkan di bucket atau akun. Saat Anda mengaktifkan

Blokir Akses Publik, temuan publik diselesaikan dan IAM Access Analyzer menghasilkan temuan lintas akun untuk titik akses lintas akun.

Setelan Amazon S3 Blokir Akses Publik mengesampingkan kebijakan bucket yang diterapkan ke bucket. Pengaturan tersebut juga membatalkan kebijakan titik akses yang berlaku pada titik akses bucket. IAM Access Analyzer menganalisis setelan Blokir Akses Publik di tingkat bucket setiap kali kebijakan berubah. Namun, ini mengevaluasi pengaturan Blokir Akses Publik di tingkat akun hanya sekali setiap 6 jam. Ini berarti bahwa IAM Access Analyzer mungkin tidak menghasilkan atau menyelesaikan temuan untuk akses publik ke bucket hingga 6 jam. Misalnya, jika Anda memiliki kebijakan bucket yang memungkinkan akses publik, IAM Access Analyzer akan menghasilkan temuan untuk akses tersebut. Jika Anda kemudian mengaktifkan Blokir Akses Publik untuk memblokir semua akses publik ke bucket di tingkat akun, IAM Access Analyzer tidak menyelesaikan temuan kebijakan bucket hingga 6 jam, meskipun semua akses publik ke bucket diblokir. Resolusi temuan publik untuk jalur akses lintas akun juga dapat memakan waktu hingga 6 jam setelah Anda mengaktifkan Blokir Akses Publik di tingkat akun. Perubahan pada kebijakan kontrol sumber daya (RCP) tanpa perubahan pada kebijakan bucket tidak memicu pemindaian ulang bucket yang dilaporkan dalam temuan. IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya, yaitu dalam waktu 24 jam.

Untuk jalur akses Multi-wilayah, IAM Access Analyzer menggunakan kebijakan yang ditetapkan untuk menghasilkan temuan. IAM Access Analyzer mengevaluasi perubahan pada titik akses Multi-wilayah setiap 6 jam sekali. Ini berarti IAM Access Analyzer tidak menghasilkan atau menyelesaikan temuan hingga 6 jam, meskipun Anda membuat atau menghapus jalur akses Multi-wilayah, atau memperbarui kebijakan untuk itu.

Ember direktori Layanan Penyimpanan Sederhana Amazon

Bucket direktori Amazon S3 mengatur data secara hierarkis ke dalam direktori sebagai lawan dari struktur penyimpanan datar dari bucket tujuan umum, yang direkomendasikan untuk beban kerja atau aplikasi yang kritis terhadap kinerja. Untuk penganalisis akses eksternal, IAM Access Analyzer menganalisis kebijakan bucket direktori, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses bucket direktori.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan saat prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke bucket direktori Amazon S3 yang ditentukan.

Bucket direktori Amazon S3 juga mendukung jalur akses, yang memberlakukan izin dan kontrol jaringan yang berbeda untuk semua permintaan yang dibuat ke bucket direktori melalui jalur akses. Setiap titik akses dapat memiliki kebijakan titik akses yang berfungsi bersama dengan kebijakan bucket yang dilampirkan ke bucket direktori yang mendasarinya. Dengan titik akses untuk bucket direktori, Anda dapat membatasi akses ke awalan tertentu, tindakan API, atau virtual private cloud (VPC).

> **Note**
>
> IAM Access Analyzer tidak menganalisis kebijakan titik akses yang dilampirkan ke jalur akses lintas akun karena titik akses dan kebijakannya berada di luar akun penganalisis. IAM Access Analyzer menghasilkan temuan publik saat bucket mendelegasikan akses ke titik akses lintas akun dan Blokir Akses Publik tidak diaktifkan di bucket atau akun. Saat Anda mengaktifkan Blokir Akses Publik, temuan publik diselesaikan dan IAM Access Analyzer menghasilkan temuan lintas akun untuk titik akses lintas akun.

Untuk informasi selengkapnya tentang bucket direktori Amazon S3, lihat Bekerja dengan bucket direktori di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

AWS Identity and Access Management peran

Untuk peran IAM, IAM Access Analyzer menganalisis kebijakan kepercayaan. Dalam kebijakan kepercayaan peran, Anda menetapkan penanggung jawab yang Anda percayai untuk mengasumsikan peran tersebut. Kebijakan kepercayaan peran adalah kebijakan wajib berbasis sumber daya yang melekat pada peran di IAM. IAM Access Analyzer menghasilkan temuan untuk peran dalam zona kepercayaan yang dapat diakses oleh entitas eksternal yang berada di luar zona kepercayaan Anda.

> **Note**
>
> Peran IAM adalah sumber daya global. Jika kebijakan kepercayaan peran memberikan akses ke entitas eksternal, IAM Access Analyzer akan menghasilkan temuan di setiap Wilayah yang diaktifkan.

AWS Key Management Service kunci

Untuk AWS KMS keys, IAM Access Analyzer menganalisis kebijakan utama dan hibah yang diterapkan pada kunci. IAM Access Analyzer menghasilkan temuan jika kebijakan atau hibah kunci memungkinkan entitas eksternal untuk mengakses kunci. Misalnya, jika Anda menggunakan kunci CallerAccount kondisi kms: dalam pernyataan kebijakan untuk mengizinkan akses ke semua pengguna di AWS akun tertentu, dan Anda menentukan akun selain akun saat ini (zona kepercayaan untuk penganalisis saat ini), IAM Access Analyzer menghasilkan temuan. Untuk mempelajari selengkapnya tentang kunci AWS KMS kondisi dalam pernyataan kebijakan IAM, lihat Kunci AWS KMS Kondisi.

Ketika IAM Access Analyzer menganalisis kunci KMS, ia membaca metadata kunci, seperti kebijakan kunci dan daftar hibah. Jika kebijakan kunci tidak mengizinkan peran IAM Access Analyzer untuk membaca metadata kunci, pencarian kesalahan Access Denied akan dihasilkan. Misalnya, jika pernyataan kebijakan contoh berikut adalah satu-satunya kebijakan yang diterapkan pada kunci, itu menghasilkan pencarian kesalahan Access ditolak di IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Karena pernyataan ini hanya mengizinkan peran bernama Admin dari AWS akun 111122223333 untuk mengakses kunci, pencarian kesalahan Access Denied dihasilkan karena IAM Access Analyzer tidak dapat sepenuhnya menganalisis kunci. Temuan kesalahan ditampilkan dalam teks merah di tabel Temuan. Temuan ini terlihat mirip dengan yang berikut ini.

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
```

```
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Saat Anda membuat kunci KMS, izin yang diberikan untuk mengakses kunci bergantung pada cara Anda membuat kunci. Jika Anda menerima pencarian kesalahan Access Denied untuk sumber daya kunci, terapkan pernyataan kebijakan berikut ke sumber daya kunci untuk memberikan izin IAM Access Analyzer untuk mengakses kunci.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Setelah Anda menerima temuan Akses Ditolak untuk sumber daya kunci KMS, kemudian menyelesaikan temuan tersebut dengan memperbarui kebijakan kunci, temuan diperbarui ke status Terselesaikan. Jika ada pernyataan kebijakan atau kunci izin yang mengizinkan kunci ke entitas eksternal, Anda mungkin melihat temuan tambahan untuk sumber daya kunci.

AWS Lambda fungsi dan lapisan

Untuk AWS Lambda fungsi, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memberikan akses ke fungsi ke entitas eksternal. Dengan Lambda, Anda dapat melampirkan kebijakan berbasis sumber daya unik ke fungsi, versi, alias, dan lapisan. IAM Access Analyzer melaporkan akses eksternal berdasarkan kebijakan berbasis sumber daya yang melekat pada fungsi dan lapisan. IAM Access Analyzer tidak melaporkan akses eksternal berdasarkan kebijakan berbasis sumber daya yang dilampirkan ke alias dan versi tertentu yang dipanggil menggunakan ARN yang memenuhi syarat.

Untuk informasi selengkapnya, lihat Menggunakan kebijakan berbasis sumber daya untuk Lambda dan Menggunakan versi di Panduan Pengembang. AWS Lambda

Antrean Amazon Simple Queue Service

Untuk antrian Amazon SQS, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses antrian.

AWS Secrets Manager rahasia

Untuk AWS Secrets Manager rahasia, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses rahasia.

Topik Amazon Simple Notification Service

IAM Access Analyzer menganalisis kebijakan berbasis sumber daya yang dilampirkan pada topik Amazon SNS, termasuk pernyataan kondisi dalam kebijakan yang memungkinkan akses eksternal ke suatu topik. Anda dapat mengizinkan akun eksternal untuk melakukan tindakan Amazon SNS seperti berlangganan dan menerbitkan topik melalui kebijakan berbasis sumber daya. Topik Amazon SNS dapat diakses secara eksternal jika kepala sekolah dari akun di luar zona kepercayaan Anda dapat melakukan operasi pada topik tersebut. Saat Anda memilih `Everyone` kebijakan saat membuat topik Amazon SNS, Anda membuat topik tersebut dapat diakses oleh publik. `AddPermission`adalah cara lain untuk menambahkan kebijakan berbasis sumber daya ke topik Amazon SNS yang memungkinkan akses eksternal.

Cuplikan volume Amazon Elastic Block Store

Snapshot volume Amazon Elastic Block Store tidak memiliki kebijakan berbasis sumber daya. Snapshot dibagikan melalui izin berbagi Amazon EBS. Untuk snapshot volume Amazon EBS, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot. Snapshot volume Amazon EBS dapat dibagikan dengan akun eksternal saat dienkripsi. Snapshot volume yang tidak terenkripsi dapat dibagikan dengan akun eksternal dan memberikan akses publik. Pengaturan berbagi ada di `CreateVolumePermissions` atribut snapshot. Saat pelanggan melihat pratinjau akses eksternal snapshot Amazon EBS, mereka dapat menentukan kunci enkripsi sebagai indikator bahwa snapshot dienkripsi, mirip dengan cara pratinjau IAM Access Analyzer menangani rahasia Secrets Manager.

Cuplikan DB Layanan Amazon Relational Database Service

Snapshot Amazon RDS DB tidak memiliki kebijakan berbasis sumber daya. Snapshot DB dibagikan melalui izin database Amazon RDS, dan hanya snapshot DB manual yang dapat dibagikan. Untuk penganalisis akses eksternal, IAM Access Analyzer menganalisis daftar kontrol akses yang

memungkinkan entitas eksternal mengakses snapshot Amazon RDS DB. Snapshot DB yang tidak terenkripsi dapat bersifat publik. Snapshot DB terenkripsi tidak dapat dibagikan secara publik, tetapi dapat dibagikan dengan hingga 20 akun lainnya. Untuk informasi selengkapnya, lihat Membuat snapshot DB. IAM Access Analyzer menganggap kemampuan untuk mengekspor snapshot manual database (misalnya, ke bucket Amazon S3) sebagai akses tepercaya.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan ketika prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke snapshot Amazon RDS DB yang ditentukan.

> **Note**
>
> IAM Access Analyzer tidak mengidentifikasi akses publik atau lintas akun yang dikonfigurasi langsung pada database itu sendiri. IAM Access Analyzer hanya mengidentifikasi temuan untuk akses publik atau lintas akun yang dikonfigurasi pada snapshot Amazon RDS DB.

Cuplikan cluster DB Layanan Relational Database Service Amazon

Snapshot klaster Amazon RDS DB tidak memiliki kebijakan berbasis sumber daya. Snapshot dibagikan melalui izin cluster Amazon RDS DB. Untuk penganalisis akses eksternal, IAM Access Analyzer menganalisis daftar kontrol akses yang memungkinkan entitas eksternal mengakses snapshot kluster Amazon RDS DB. Snapshot cluster yang tidak terenkripsi dapat bersifat publik. Snapshot kluster terenkripsi tidak dapat dibagikan secara publik. Snapshot cluster yang tidak terenkripsi dan terenkripsi dapat dibagikan dengan hingga 20 akun lainnya. Untuk informasi selengkapnya, lihat Membuat snapshot cluster DB. IAM Access Analyzer menganggap kemampuan untuk mengekspor snapshot cluster DB (misalnya, ke bucket Amazon S3) sebagai akses tepercaya.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan ketika prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke snapshot kluster Amazon RDS DB yang ditentukan.

> **Note**
>
> Temuan IAM Access Analyzer tidak mencakup pemantauan bagian mana pun dari cluster Amazon RDS DB dan klon dengan yang lain atau organisasi yang menggunakan. Akun AWS AWS Resource Access Manager IAM Access Analyzer hanya mengidentifikasi temuan untuk akses publik atau lintas akun yang dikonfigurasi pada snapshot klaster Amazon RDS DB.

Repositori Registri Wadah Elastis Amazon

Untuk repositori Amazon ECR, IAM Access Analyzer menganalisis kebijakan berbasis sumber daya, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses repositori (mirip dengan jenis sumber daya lain seperti topik Amazon SNS dan sistem file Amazon EFS). Untuk repositori Amazon ECR, prinsipal harus memiliki izin untuk `ecr:GetAuthorizationToken` melalui kebijakan berbasis identitas agar dianggap tersedia secara eksternal.

Sistem file Amazon Elastic File System

Untuk sistem file Amazon EFS, IAM Access Analyzer menganalisis kebijakan, termasuk pernyataan kondisi dalam kebijakan, yang memungkinkan entitas eksternal mengakses sistem file. Sistem file Amazon EFS dapat diakses secara eksternal jika prinsipal dari akun di luar zona kepercayaan Anda dapat melakukan operasi pada sistem file tersebut. Akses didefinisikan oleh kebijakan sistem file yang menggunakan IAM, dan bagaimana sistem file dipasang. Misalnya, memasang sistem file Amazon EFS Anda di akun lain dianggap dapat diakses secara eksternal, kecuali akun tersebut ada di organisasi Anda dan Anda telah mendefinisikan organisasi sebagai zona kepercayaan Anda. Jika Anda memasang sistem file dari cloud pribadi virtual dengan subnet publik, sistem file dapat diakses secara eksternal. Saat Anda menggunakan Amazon EFS dengan AWS Transfer Family, permintaan akses sistem file yang diterima dari server Transfer Family yang dimiliki oleh akun berbeda dari sistem file akan diblokir jika sistem file mengizinkan akses publik.

Aliran Amazon DynamoDB

Untuk penganalisis akses eksternal, IAM Access Analyzer menghasilkan temuan jika kebijakan DynamoDB mengizinkan setidaknya satu tindakan lintas akun yang memungkinkan entitas eksternal mengakses aliran DynamoDB. Untuk informasi selengkapnya tentang tindakan lintas akun yang didukung untuk DynamoDB, lihat Tindakan IAM yang didukung oleh kebijakan berbasis sumber daya di Panduan Pengembang Amazon DynamoDB.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan ketika prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke aliran DynamoDB tertentu.

Tabel Amazon DynamoDB

Untuk penganalisis akses eksternal, IAM Access Analyzer menghasilkan temuan untuk tabel DynamoDB jika kebijakan DynamoDB mengizinkan setidaknya satu tindakan lintas akun yang memungkinkan entitas eksternal mengakses tabel atau indeks DynamoDB. Untuk informasi

selengkapnya tentang tindakan lintas akun yang didukung untuk DynamoDB, lihat Tindakan IAM yang didukung oleh kebijakan berbasis sumber daya di Panduan Pengembang Amazon DynamoDB.

Untuk penganalisis akses internal, IAM Access Analyzer menghasilkan temuan ketika prinsipal (pengguna atau peran) dalam organisasi atau akun Anda memiliki akses ke tabel DynamoDB tertentu.

Administrator yang didelegasikan untuk IAM Access Analyzer

Jika Anda mengonfigurasi AWS Identity and Access Management Access Analyzer di akun AWS Organizations manajemen, Anda dapat menambahkan akun anggota di organisasi sebagai administrator yang didelegasikan untuk mengelola IAM Access Analyzer untuk organisasi Anda. Administrator yang didelegasikan memiliki izin untuk membuat dan mengelola penganalisis dalam organisasi. Hanya akun manajemen yang dapat menambahkan administrator yang didelegasikan.

Administrator yang didelegasikan untuk IAM Access Analyzer adalah akun anggota dalam organisasi yang memiliki izin untuk membuat dan mengelola penganalisis yang menganalisis akses di seluruh organisasi. Hanya akun manajemen yang dapat menambahkan, menghapus, atau mengubah administrator yang telah didelegasikan.

Jika Anda menambahkan administrator yang didelegasikan, Anda dapat berganti ke akun lain untuk administrator yang didelegasikan. Ketika Anda melakukannya, akun administrator yang didelegasikan sebelumnya kehilangan izin untuk semua penganalisis yang dibuat menggunakan akun tersebut untuk menganalisis akses di seluruh organisasi. Penganalisis ini berpindah ke status dinonaktifkan dan tidak lagi menghasilkan temuan baru atau memperbarui temuan yang ada. Temuan yang ada untuk penganalisis ini juga tidak lagi dapat diakses. Anda dapat mengaksesnya di masa mendatang dengan mengonfigurasi akun tersebut sebagai administrator yang didelagasikan. Jika Anda tahu bahwa Anda tidak akan menggunakan akun yang sama sebagai administrator yang didelegasikan, pertimbangkan untuk menghapus penganalisis sebelum mengubah administrator yang didelegasikan. Ini akan menghapus semua temuan yang dihasilkan. Ketika administrator baru yang dihilangkan membuat penganalisis baru, maka akan menghasilkan instans baru dari temuan yang sama. Anda tidak kehilangan temuan apa pun, mereka hanya dihasilkan untuk penganalisis baru di akun lain. Dan Anda dapat terus mengakses temuan untuk organisasi menggunakan akun manajemen organisasi, yang juga memiliki izin administrator. Administrator baru yang didelegasikan harus membuat penganalisis baru untuk IAM Access Analyzer untuk mulai memantau sumber daya di organisasi Anda.

Jika administrator yang didelegasikan meninggalkan AWS organisasi, hak istimewa administrasi yang didelegasikan akan dihapus dari akun. Semua penganalisis di akun dengan organisasi sebagai zona

kepercayaan berpindah ke keadaan dinonaktifkan. Temuan yang ada untuk penganalisis ini juga tidak lagi dapat diakses.

Saat pertama kali Anda mengonfigurasi penganalisis di akun manajemen, Anda dapat memilih Tambahkan administrator yang didelegasikan di halaman pengaturan Analyzer di konsol IAM Access Analyzer.

> **Note**
>
> IAM Access Analyzer mengenakan biaya untuk penganalisis akses yang tidak digunakan berdasarkan jumlah peran IAM dan pengguna yang dianalisis per penganalisis per bulan. Jika Anda membuat penganalisis akses yang tidak digunakan di akun manajemen dan akun administrator yang didelegasikan, Anda akan dikenakan biaya untuk kedua penganalisis akses yang tidak digunakan. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Setelah Anda mengubah administrator yang didelegasikan, administrator yang baru harus membuat penganalisis untuk mulai memantau akses ke sumber daya di organisasi Anda.

Tambahkan administrator yang didelegasikan untuk IAM Access Analyzer

Jika Anda mengonfigurasi AWS Identity and Access Management Access Analyzer di akun AWS Organizations manajemen, Anda dapat menambahkan akun anggota di organisasi sebagai administrator yang didelegasikan untuk mengelola IAM Access Analyzer untuk organisasi Anda. Administrator yang didelegasikan memiliki izin untuk membuat dan mengelola penganalisis dalam organisasi. Hanya akun manajemen yang dapat menambahkan administrator yang didelegasikan.

Untuk menambahkan administrator yang didelegasikan dengan menggunakan konsol

1. Masuk ke AWS konsol menggunakan akun manajemen untuk organisasi Anda.
2. Buka konsol IAM di https://console.aws.amazon.com/iam/.
3. Di bawah Access Analyzer, pilih Pengaturan Analyzer.
4. Pilih Tambahkan administrator yang didelegasikan.
5. Di bidang Administrator yang didelegasikan, masukkan Akun AWS nomor akun anggota organisasi untuk membuat administrator yang didelegasikan.

   Akun tersebut harus menjadi anggota dari organisasi Anda.

6. Pilih Simpan perubahan.

Untuk menambahkan administrator yang didelegasikan menggunakan AWS CLI atau AWS SDKs

Saat Anda membuat penganalisis untuk menganalisis akses di seluruh organisasi dalam akun administrator yang didelegasikan menggunakan AWS CLI, AWS API (menggunakan AWS SDKs) atau CloudFormation, Anda harus menggunakannya AWS Organizations APIs untuk mengaktifkan akses layanan untuk IAM Access Analyzer dan mendaftarkan akun anggota sebagai administrator yang didelegasikan.

1. Aktifkan akses layanan tepercaya untuk IAM Access Analyzer di. AWS Organizations Lihat Cara Mengaktifkan atau Menonaktifkan Akses Tepercaya di Panduan AWS Organizations Pengguna.
2. Daftarkan akun anggota yang valid dari AWS organisasi Anda sebagai administrator yang didelegasikan menggunakan operasi AWS Organizations `RegisterDelegatedAdministrator`API atau `register-delegated-administrator` AWS CLI perintah.

Aturan arsip

Aturan arsip secara otomatis mengarsipkan temuan baru yang memenuhi kriteria yang Anda tentukan saat membuat aturan. Anda juga dapat menerapkan aturan arsip secara retroaktif untuk mengarsipkan temuan yang ada yang memenuhi kriteria aturan arsip. Misalnya, Anda dapat membuat aturan arsip untuk secara otomatis mengarsipkan temuan apa pun untuk bucket Amazon S3 tertentu yang dapat Anda akses secara teratur. Atau jika Anda memberikan akses ke beberapa sumber daya ke prinsip tertentu, Anda dapat membuat aturan yang secara otomatis mengarsipkan setiap temuan baru yang dihasilkan untuk akses yang diberikan ke prinsip tersebut. Hal ini memungkinkan Anda fokus hanya pada temuan aktif yang mungkin menunjukkan risiko keamanan.

Saat Anda membuat aturan arsip, hanya temuan baru yang cocok dengan kriteria aturan yang diarsipkan secara otomatis. Temuan yang ada tidak diarsipkan secara otomatis. Saat membuat aturan, Anda dapat menyertakan hingga 20 nilai per kriteria di aturan. Untuk daftar kunci filter yang dapat Anda gunakan untuk membuat atau memperbarui aturan arsip, lihat Kunci filter IAM Access Analyzer.

> **Note**
>
> Saat Anda membuat atau mengedit aturan arsip, IAM Access Analyzer tidak memvalidasi nilai yang Anda sertakan dalam filter untuk aturan tersebut. Misalnya, jika Anda menambahkan aturan untuk mencocokkan Akun AWS, IAM Access Analyzer menerima nilai apa pun di bidang, meskipun itu bukan nomor akun yang valid AWS .

Untuk membuat aturan arsip

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Access analyzer, lalu pilih Pengaturan Analyzer.
3. Di bagian Analyzers, pilih analyzer yang ingin Anda buat aturan arsipnya.
4. Pada tab Aturan arsip, pilih Buat aturan arsip.
5. Masukkan nama untuk aturan jika Anda ingin mengubah nama default.
6. Di bagian Aturan, di bawah Kriteria, pilih properti yang sesuai dengan aturan.
7. Pilih kondisi untuk nilai properti, seperti Contains, Is, atau Not Equals.

   Operator yang tersedia tergantung pada properti yang Anda pilih.
8. Secara opsional, tambahkan nilai tambahan untuk properti, atau tambahkan kriteria tambahan untuk aturan. Untuk temuan akses eksternal, untuk memastikan bahwa aturan Anda tidak akan mengarsipkan temuan baru untuk akses publik, Anda juga dapat menyertakan kriteria Akses publik dan mengaturnya ke false.

   Untuk menambahkan nilai lain bagi kriteria, pilih Tambahkan nilai lain. Untuk menambahkan kriteria lain untuk aturan, pilih Tambahkan kriteria.
9. Setelah selesai menambahkan kriteria dan nilai, pilih Buat aturan untuk menerapkan aturan hanya pada temuan baru. Pilih Buat dan arsipkan temuan aktif untuk mengarsipkan temuan baru dan temuan yang ada berdasarkan kriteria aturan. Di bagian Hasil, Anda dapat meninjau daftar temuan aktif yang diterapkan aturan arsip.

Misalnya, untuk membuat aturan untuk temuan akses eksternal yang secara otomatis mengarsipkan temuan apa pun untuk bucket Amazon S3: pilih Jenis sumber daya, lalu pilih Is untuk kondisi tersebut. Selanjutnya pilih bucket S3 dari daftar Nilai.

Untuk membuat aturan untuk temuan akses yang tidak digunakan yang secara otomatis mengarsipkan temuan apa pun untuk akun tertentu: pilih Akun Pemilik Sumber Daya, lalu pilih Sama dengan kondisi tersebut. Ketik Akun AWS ID di kotak teks Nilai.

Lanjutkan untuk menentukan kriteria untuk menyesuaikan aturan yang sesuai untuk lingkungan Anda, lalu pilih Buat aturan.

Jika Anda membuat aturan baru dan menambahkan beberapa kriteria, Anda dapat menghapus satu kriteria dari aturan dengan memilih Hapus kriteria ini. Anda dapat menghapus nilai tambah untuk kriteria dengan memilih Hapus nilai.

Untuk mengedit aturan arsip

1. Pilih nama aturan yang akan diedit di kolom Nama.

   Anda hanya dapat mengedit satu aturan arsip dalam satu waktu.
2. Tambahkan kriteria baru atau hapus kriteria dan nilai yang ada untuk setiap kriteria.
3. Pilih Simpan perubahan untuk menerapkan aturan hanya pada temuan baru. Pilih Simpan dan arsipkan temuan aktif untuk mengarsipkan temuan baru dan temuan yang ada berdasarkan kriteria aturan.

Untuk menghapus aturan arsip

1. Pilih kotak centang untuk aturan yang ingin Anda hapus.
2. Pilih Hapus.
3. Jenis **`delete`** di dialog konfirmasi Hapus aturan arsip, lalu pilih Hapus.

Aturan hanya dihapus dari penganalisis di Wilayah saat ini. Anda harus menghapus aturan arsip secara terpisah untuk setiap penganalisis yang Anda buat di Wilayah lain.

Pemantauan AWS Identity and Access Management Access Analyzer dengan Amazon EventBridge

Gunakan informasi dalam topik ini untuk mempelajari cara memantau temuan IAM Access Analyzer dan mengakses pratinjau dengan Amazon. EventBridge EventBridge adalah versi baru dari Amazon CloudWatch Events.

Peristiwa temuan

IAM Access Analyzer mengirimkan acara EventBridge untuk setiap temuan yang dihasilkan, untuk perubahan status temuan yang ada, dan saat temuan dihapus. Untuk menerima temuan dan pemberitahuan tentang temuan, Anda harus membuat aturan acara di Amazon EventBridge. Saat Anda membuat aturan peristiwa, Anda juga dapat menetapkan tindakan target yang akan dipicu berdasarkan aturan tersebut. Misalnya, Anda dapat membuat aturan acara yang memicu topik Amazon SNS saat peristiwa untuk temuan baru diterima dari IAM Access Analyzer. Rincian tentang kebijakan kontrol sumber daya (RCP) tersedia di bagian detail acara.

Akses peristiwa pratinjau

IAM Access Analyzer mengirimkan acara EventBridge untuk setiap pratinjau akses dan mengubah statusnya. Ini termasuk peristiwa ketika pratinjau akses pertama kali dibuat (status Membuat), ketika pratinjau akses selesai (status Selesai), atau ketika pembuatan pratinjau akses gagal (status Gagal). Untuk menerima pemberitahuan tentang pratinjau akses, Anda harus membuat aturan peristiwa di EventBridge. Saat Anda membuat aturan peristiwa, Anda dapat menetapkan tindakan target yang akan dipicu berdasarkan aturan tersebut. Misalnya, Anda dapat membuat aturan peristiwa yang memicu topik Amazon SNS saat peristiwa untuk pratinjau akses lengkap diterima dari IAM Access Analyzer.

Frekuensi pemberitahuan peristiwa

IAM Access Analyzer mengirimkan peristiwa untuk temuan dan temuan baru dengan pembaruan status EventBridge dalam waktu sekitar satu jam sejak peristiwa terjadi di akun Anda. IAM Access Analyzer juga mengirimkan peristiwa ke EventBridge saat temuan yang diselesaikan dihapus karena periode retensi telah kedaluwarsa. Untuk temuan yang dihapus karena penganalisis yang menghasilkannya dihapus, acara dikirim ke EventBridge sekitar 24 jam setelah penganalisis dihapus. Saat temuan dihapus, status temuan tidak berubah. Sebaliknya, atribut `isDeleted` diatur menjadi `true`. IAM Access Analyzer juga mengirimkan acara untuk pratinjau akses yang baru dibuat dan perubahan status pratinjau akses ke. EventBridge

Contoh peristiwa temuan akses eksternal

Berikut ini adalah contoh IAM Access Analyzer external access finding event yang dikirim ke. EventBridge Yang `id` tercantum adalah ID untuk acara di EventBridge. Untuk mempelajari lebih lanjut, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Di objek `detail`, nilai untuk atribut `accountId` dan `region` merujuk pada akun dan wilayah yang dilaporkan dalam temuan. Atribut `isDeleted` menunjukkan jika peristiwa berasal dari temuan yang dihapus. `id` adalah ID temuan. Susunan `resources` adalah susunan tunggal dengan ARN dari penganalisis yang menghasilkan temuan.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAM Access Analyzer juga mengirimkan peristiwa EventBridge untuk temuan kesalahan. Temuan kesalahan adalah temuan yang dihasilkan ketika IAM Access Analyzer tidak dapat menganalisis

sumber daya. Peristiwa untuk temuan kesalahan meliputi `error` sebagaimana ditunjukkan dalam contoh berikut.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Contoh peristiwa temuan akses internal

Berikut ini adalah contoh IAM Access Analyzer internal access finding event yang dikirim ke. EventBridge Yang `id` tercantum adalah ID untuk acara di EventBridge. Untuk mempelajari lebih lanjut, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Dalam `detail` objek, nilai untuk `accountId` dan `principalOwnerAccount` atribut mengacu pada akun prinsipal yang dilaporkan dalam temuan. Atribut `isDeleted` menunjukkan jika peristiwa berasal dari temuan yang dihapus. `id` adalah ID temuan. `resource`Itu adalah ARN dari penganalisis yang menghasilkan temuan.

```
{
```

```
    "version": "0",
    "id": "b45c3678-c278-b593-6121-c155259ce1b5",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2025-04-08T19:42:49Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:access-analyzer:us-east-1:111122223333:analyzer/testAnalyzer"
    ],
    "detail": {
        "accessType": "INTRA_ACCOUNT",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2025-04-08T03:18:43.509465073Z",
        "condition": {},
        "createdAt": "2025-04-07T21:33:49.914099224Z",
        "id": "11111111-2222-4444-aaaa-333333333333",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "principal": {
            "AWS": "arn:aws:iam::111122223333:role/MyRole_6"
        },
        "principalOwnerAccount": "111122223333",
        "principalType": "IAM_ROLE",
        "resource": "arn:aws:s3:::critical-data",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "accountId": "111122223333",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "status": "ACTIVE",
        "updatedAt": "2025-04-08T03:22:12.654688231Z",
        "version": "1.0"
    }
}
```


IAM Access Analyzer juga mengirimkan peristiwa EventBridge untuk temuan kesalahan. Temuan kesalahan adalah temuan yang dihasilkan ketika IAM Access Analyzer tidak dapat menganalisis sumber daya. Peristiwa untuk temuan kesalahan meliputi `error` sebagaimana ditunjukkan dalam contoh berikut.


```
{
```

```
    "version": "0",
    "id": "5a94b99b-e87d-a6a7-58c7-f47871532860",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer-test",
    "account": "444455556666",
    "time": "2025-05-07T11:57:54Z",
    "region": "us-west-2",
    "resources": ["arn:aws:access-analyzer-beta:us-west-2:444455556666:analyzer/
example-analyzer"],
    "detail": {
        "analyzedAt": "2025-03-24T19:58:52.512329448Z",
        "createdAt": "2025-03-22T03:30:46.920200692Z",
        "id": "ef573afd-12a5-4095-87a6-bf2f25109895",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "resource": "arn:aws:s3:::test-entity-88",
        "accountId": "111122223333",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "error": "ACCESS_DENIED", // can be INTERNAL_ERROR and ACCESS_DENIED
        "status": "ACTIVE",
        "updatedAt": "2025-03-24T20:09:39.176075014Z",
        "version": "1.0"
    }
}
```

Contoh temuan akses yang tidak digunakan terkait peristiwa

Berikut ini adalah contoh IAM Access Analyzer unused access finding event yang dikirim ke. EventBridge Yang `id` tercantum adalah ID untuk acara di EventBridge. Untuk mempelajari lebih lanjut, lihat Peristiwa dan Pola Peristiwa di EventBridge.

Di objek `detail`, nilai untuk atribut `accountId` dan `region` merujuk pada akun dan wilayah yang dilaporkan dalam temuan. Atribut `isDeleted` menunjukkan jika peristiwa berasal dari temuan yang dihapus. `id` adalah ID temuan.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
```

```
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

IAM Access Analyzer juga mengirimkan peristiwa EventBridge untuk temuan kesalahan. Temuan kesalahan adalah temuan yang dihasilkan ketika IAM Access Analyzer tidak dapat menganalisis sumber daya. Peristiwa untuk temuan kesalahan meliputi `error` sebagaimana ditunjukkan dalam contoh berikut.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
```

```
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Contoh peristiwa pratinjau akses

Contoh berikut menunjukkan data untuk acara pertama yang dikirim EventBridge saat Anda membuat pratinjau akses. Array `resources` adalah susunan tunggal dengan ARN penganalisis yang terkait dengan pratinjau akses. Di objek `detail`, `id` mengacu pada ID pratinjau akses dan `configuredResources` mengacu pada sumber daya yang pratinjau aksesnya dibuat. `status` adalah `Creating` dan mengacu pada status pratinjau akses. Parameter `previousStatus` tidak ditentukan karena pratinjau akses baru saja dibuat.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
```

```
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Contoh berikut menunjukkan data untuk peristiwa yang dikirim ke EventBridge untuk pratinjau akses dengan perubahan status dari `Creating` ke`Completed`. Dalam objek detail, `id` mengacu pada ID pratinjau akses. `status` dan `previousStatus` mengacu pada status pratinjau akses, ketika status sebelumnya `Creating` dan status saat ini adalah `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Contoh berikut menunjukkan data untuk peristiwa yang dikirim ke EventBridge untuk pratinjau akses dengan perubahan status dari `Creating` ke`Failed`. Di objek `detail`, `id` mengacu pada ID pratinjau akses. `status` dan `previousStatus` mengacu pada status pratinjau akses, ketika status sebelumnya `Creating` dan status saat ini adalah `Failed`. Bidang `statusReason` menyediakan kode alasan yang menunjukkan bahwa pratinjau akses gagal karena konfigurasi sumber daya yang tidak valid.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```


Membuat aturan peristiwa menggunakan konsol

Prosedur berikut menjelaskan cara membuat aturan peristiwa menggunakan konsol.

1. Buka EventBridge konsol Amazon di https://console.aws.amazon.com/events/.
2. Dengan menggunakan nilai berikut, buat EventBridge aturan yang memantau pencarian peristiwa atau mengakses peristiwa pratinjau:

    - Untuk Tipe aturan, pilih Aturan dengan pola peristiwa.
    - Untuk sumber acara, pilih Lainnya.
    - Untuk pola Acara, pilih Pola kustom (editor JSON), dan tempelkan salah satu contoh pola peristiwa berikut ke area teks:
        - Untuk membuat aturan berdasarkan peristiwa IAM Access Analyzer, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ]
}
```

- Untuk membuat aturan berdasarkan akses eksternal, akses internal, atau peristiwa temuan akses yang tidak digunakan, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding",
    "Internal Access Finding",
    "Unused Access Finding for IAM entities"
  ]
}
```

- Untuk membuat aturan hanya berdasarkan peristiwa temuan akses eksternal, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

- Untuk membuat aturan hanya berdasarkan peristiwa temuan akses internal, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Internal Access Finding"
```

```
  ]
}
```

- Untuk membuat aturan hanya berdasarkan peristiwa temuan akses yang tidak digunakan, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

- Untuk membuat aturan berdasarkan peristiwa pratinjau akses, gunakan contoh pola berikut:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Untuk jenis Target, pilih AWS layanan, dan untuk Pilih target, pilih target seperti topik atau AWS Lambda fungsi Amazon SNS. Target terpicu saat peristiwa diterima yang sesuai dengan pola peristiwa yang ditentukan dalam aturan.

Untuk mempelajari selengkapnya tentang cara membuat aturan, lihat Membuat EventBridge aturan Amazon yang bereaksi terhadap peristiwa di Panduan EventBridge Pengguna Amazon.

Membuat aturan peristiwa menggunakan CLI

1. Gunakan yang berikut ini untuk membuat aturan untuk Amazon EventBridge menggunakan AWS CLI. Ganti nama aturan *TestRule* dengan nama untuk aturan Anda.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"]}"
```

2. Anda dapat menyesuaikan aturan untuk memicu tindakan target hanya bagi sebagian temuan yang dihasilkan, seperti temuan dengan atribut spesifik. Contoh berikut ini menunjukkan cara membuat aturan yang memicu tindakan target hanya untuk temuan dengan status Aktif.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":
[\"ACTIVE\"]}}"
```

   Contoh berikut menunjukkan cara membuat aturan yang memicu tindakan target hanya untuk pratinjau akses dengan status from to. `Creating` `Completed`

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Untuk menetapkan fungsi Lambda sebagai target untuk aturan yang Anda buat, gunakan perintah contoh berikut. Ganti Wilayah dan nama fungsi dalam ARN yang sesuai dengan lingkungan Anda.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Tambahkan izin yang diperlukan untuk memunculkan target aturan. Contoh berikut menunjukkan cara memberikan izin untuk fungsi Lambda, mengikuti contoh sebelumnya.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Integrasikan IAM Access Analyzer dengan AWS Security Hub CSPM

AWS Security Hub CSPMmemberikan pandangan komprehensif tentang keadaan keamanan Anda di seluruh AWS. Ini membantu Anda menilai lingkungan Anda terhadap standar industri keamanan dan praktik terbaik. Security Hub CSPM mengumpulkan data keamanan dari seluruh layanan Akun AWS, dan produk mitra pihak ketiga yang didukung. Kemudian menganalisis tren keamanan Anda dan mengidentifikasi masalah keamanan prioritas tertinggi.

Saat Anda mengintegrasikan IAM Access Analyzer dengan Security Hub CSPM, Anda dapat mengirim temuan dari IAM Access Analyzer ke Security Hub CSPM. Security Hub CSPM kemudian dapat memasukkan temuan-temuan tersebut dalam analisisnya tentang postur keamanan Anda secara keseluruhan.

Daftar Isi

- Bagaimana IAM Access Analyzer mengirimkan temuan ke Security Hub CSPM
  - Jenis temuan yang dikirim IAM Access Analyzer
  - Latensi untuk mengirim temuan
  - Mencoba lagi saat Security Hub CSPM tidak tersedia
  - Memperbarui temuan yang ada di Security Hub CSPM
- Melihat temuan IAM Access Analyzer di Security Hub CSPM
  - Menafsirkan IAM Access Analyzer menemukan nama di Security Hub CSPM
- Temuan khas dari IAM Access Analyzer
- Mengaktifkan dan mengonfigurasi integrasi
- Bagaimana cara menghentikan pengiriman temuan

Bagaimana IAM Access Analyzer mengirimkan temuan ke Security Hub CSPM

Di Security Hub CSPM, masalah keamanan dilacak sebagai temuan. Beberapa temuan berasal dari masalah yang terdeteksi oleh pihak lain Layanan AWS atau oleh mitra pihak ketiga. Security Hub CSPM juga memiliki seperangkat aturan yang digunakan untuk mendeteksi masalah keamanan dan menghasilkan temuan.

Security Hub CSPM menyediakan alat untuk mengelola temuan dari seluruh sumber ini. Anda dapat melihat dan memfilter daftar temuan dan melihat informasi rinci tentang setiap temuan. Untuk informasi lebih lanjut, lihat Melihat temuan dalam Panduan Pengguna AWS Security Hub . Anda juga dapat melacak status investigasi terhadap temuan. Untuk informasi lebih lanjut, lihat Mengambil tindakan pada temuan dalam Panduan Pengguna AWS Security Hub .

Semua temuan di Security Hub CSPM menggunakan format JSON standar yang disebut AWS Security Finding Format (ASFF). ASFF mencakup detail tentang sumber masalah, sumber daya yang terpengaruh, dan status temuan saat ini. Untuk informasi lebih lanjut, lihat AWS Security Finding Format (ASFF) di Panduan Pengguna AWS Security Hub .

AWS Identity and Access Management Access Analyzer adalah salah satu Layanan AWS yang mengirimkan temuan ke Security Hub CSPM. Untuk akses yang tidak digunakan, IAM Access Analyzer mendeteksi akses yang tidak terpakai yang diberikan kepada pengguna atau peran IAM dan menghasilkan temuan untuk masing-masing peran tersebut. IAM Access Analyzer kemudian mengirimkan temuan ini ke Security Hub CSPM.

Untuk akses eksternal, IAM Access Analyzer mendeteksi pernyataan kebijakan yang memungkinkan akses publik atau akses lintas akun ke prinsipal eksternal pada sumber daya yang didukung di organisasi atau akun Anda. IAM Access Analyzer menghasilkan temuan untuk akses publik dan mengirimkannya ke Security Hub CSPM. Untuk akses lintas akun, IAM Access Analyzer mengirimkan satu temuan untuk satu prinsipal eksternal sekaligus ke Security Hub CSPM. Jika ada beberapa temuan lintas akun di IAM Access Analyzer, Anda harus menyelesaikan temuan CSPM Security Hub untuk prinsipal eksternal tunggal sebelum IAM Access Analyzer menyediakan temuan lintas akun berikutnya. Untuk daftar lengkap prinsipal eksternal dengan akses lintas akun di luar zona kepercayaan untuk penganalisis, Anda harus melihat temuan di IAM Access Analyzer. Rincian tentang kebijakan kontrol sumber daya (RCP) tersedia di bagian detail sumber daya.

Jenis temuan yang dikirim IAM Access Analyzer

IAM Access Analyzer mengirimkan temuan ke Security Hub CSPM menggunakan AWS Security Finding Format (ASFF). Dalam ASFF, bidang `Types` menyediakan jenis temuan. Temuan dari IAM Access Analyzer dapat memiliki nilai berikut untuk. `Types`

- Temuan akses eksternal - Effects/Data Paparan /Akses Eksternal Diberikan
- Temuan akses eksternal — Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/Keamanan Akses Terbaik Diberikan Practices/External
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan Konfigurasi/Izin Terbaik AWS Keamanan Practices/Unused
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/ Keamanan Peran IAM Terbaik Practices/Unused
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/ Keamanan Kata Sandi Pengguna IAM Terbaik Practices/Unused
- Temuan akses yang tidak digunakan — Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/ Keamanan Kunci Akses Pengguna Practices/Unused IAM Terbaik

Latensi untuk mengirim temuan

Ketika IAM Access Analyzer membuat temuan baru, biasanya dikirim ke Security Hub CSPM dalam waktu 30 menit. Namun, ada kasus yang jarang terjadi ketika IAM Access Analyzer mungkin tidak diberi tahu tentang perubahan kebijakan. Contoh:

- Perubahan pada pengaturan akses publik blok tingkat akun Amazon S3 dapat memakan waktu hingga 12 jam untuk tercermin dalam IAM Access Analyzer.
- Perubahan pada kebijakan kontrol sumber daya (RCP) tanpa perubahan pada kebijakan berbasis sumber daya tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan. IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya, yaitu dalam waktu 24 jam.
- Jika ada masalah pengiriman dengan pengiriman AWS CloudTrail log, perubahan kebijakan mungkin tidak memicu pemindaian ulang sumber daya yang dilaporkan dalam temuan.

Dalam situasi ini, IAM Access Analyzer menganalisis kebijakan baru atau yang diperbarui selama pemindaian berkala berikutnya.

Mencoba lagi saat Security Hub CSPM tidak tersedia

Jika Security Hub CSPM tidak tersedia, IAM Access Analyzer mencoba mengirimkan temuan secara berkala.

Memperbarui temuan yang ada di Security Hub CSPM

Setelah mengirim temuan ke Security Hub CSPM, IAM Access Analyzer terus mengirim pembaruan untuk mencerminkan pengamatan tambahan dari aktivitas temuan ke Security Hub CSPM. Pembaruan ini tercermin dalam temuan yang sama.

Untuk temuan akses eksternal IAM Access Analyzer mengelompokkannya per sumber daya. Di Security Hub CSPM, temuan untuk sumber daya tetap aktif jika setidaknya salah satu temuan untuk sumber daya tersebut aktif di IAM Access Analyzer. Jika semua temuan di IAM Access Analyzer untuk sumber daya diarsipkan atau diselesaikan, maka temuan CSPM Security Hub juga diarsipkan. Temuan CSPM Security Hub diperbarui ketika akses kebijakan berubah antara akses publik dan lintas akun. Pembaruan ini dapat mencakup perubahan jenis, judul, deskripsi, dan tingkat keparahan temuan.

Untuk temuan akses yang tidak digunakan, IAM Access Analyzer tidak mengelompokkannya berdasarkan sumber daya. Sebaliknya, jika temuan akses yang tidak digunakan diselesaikan di

IAM Access Analyzer, maka temuan CSPM Security Hub terkait juga diselesaikan. Temuan CSPM Security Hub diperbarui saat Anda memperbarui pengguna IAM atau peran yang menghasilkan pencarian akses yang tidak digunakan.

Melihat temuan IAM Access Analyzer di Security Hub CSPM

Untuk melihat temuan IAM Access Analyzer Anda di Security Hub CSPM, pilih Lihat temuan di bagian AWS: IAM Access Analyzer pada halaman ringkasan. Sebagai alternatif, Anda dapat memilih Temuan dari panel navigasi. Anda kemudian dapat memfilter temuan untuk hanya menampilkan AWS Identity and Access Management Access Analyzer temuan dengan memilih bidang Nama produk: dengan nilai**`IAM Access Analyzer`**.

Menafsirkan IAM Access Analyzer menemukan nama di Security Hub CSPM

AWS Identity and Access Management Access Analyzer mengirimkan temuan ke Security Hub CSPM menggunakan AWS Security Finding Format (ASFF). Dalam ASFF, Tipe menyediakan jenis temuan. Jenis ASFF menggunakan skema penamaan yang berbeda dari AWS Identity and Access Management Access Analyzer. Tabel berikut mencakup rincian tentang semua jenis ASFF yang terkait dengan AWS Identity and Access Management Access Analyzer temuan seperti yang muncul di Security Hub CSPM.

Tipe temuan ASFF	Judul pencarian CSPM Security Hub	Deskripsi
Effects/Data Exposure/ ExternalAkses Diberikan	<resource ARN> memungkinkan akses publik	Sebuah kebijakan berbasis sumber daya yang melekat pada sumber daya memungkinkan akses publik pada sumber daya untuk semua penanggung jawab eksternal.
Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/Keamanan Akses Terbaik Diberikan Practices/External	<resource ARN> memungkinkan akses lintas akun	Kebijakan berbasis sumber daya yang melekat pada sumber daya memungkinkan akses lintas akun ke pelaku eksternal di luar zona

Tipe temuan ASFF	Judul pencarian CSPM Security Hub	Deskripsi
		kepercayaan untuk penganalisis.
Pemeriksaan Perangkat Lunak dan Konfigurasi/Praktik Terbaik AWS Keamanan/Izin yang Tidak Digunakan	<resource ARN>berisi izin yang tidak digunakan	Pengguna atau peran berisi izin layanan dan tindakan yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/Keamanan Peran IAM Terbaik Practices/Unused	<resource ARN>berisi peran IAM yang tidak digunakan	Pengguna atau peran berisi peran IAM yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/Keamanan Kata Sandi Pengguna Practices/Unused IAM Terbaik	<resource ARN>berisi kata sandi pengguna IAM yang tidak digunakan	Pengguna atau peran berisi kata sandi pengguna IAM yang tidak digunakan.
Pemeriksaan Perangkat Lunak dan AWS Konfigurasi/Keamanan Kunci Akses Pengguna Practices/Unused IAM Terbaik	<resource ARN>berisi kunci akses pengguna IAM yang tidak digunakan	Pengguna atau peran berisi kunci akses pengguna IAM yang tidak digunakan.

Temuan khas dari IAM Access Analyzer

IAM Access Analyzer mengirimkan temuan ke Security Hub CSPM menggunakan AWS Security Finding Format (ASFF).

Berikut adalah contoh temuan khas dari IAM Access Analyzer untuk temuan akses eksternal.

```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::amzn-s3-demo-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
```

```
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-account
 access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-
account access from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```

Berikut adalah contoh temuan khas dari IAM Access Analyzer untuk temuan akses yang tidak digunakan.


```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
```

```
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

Mengaktifkan dan mengonfigurasi integrasi

Untuk menggunakan integrasi dengan Security Hub CSPM, Anda harus mengaktifkan Security Hub CSPM. Untuk informasi tentang cara mengaktifkan CSPM Security Hub, lihat Menyiapkan Security Hub di AWS Security Hub Panduan Pengguna.

Saat Anda mengaktifkan IAM Access Analyzer dan Security Hub CSPM, integrasi diaktifkan secara otomatis. IAM Access Analyzer segera mulai mengirimkan temuan ke Security Hub CSPM.

Bagaimana cara menghentikan pengiriman temuan

Untuk menghentikan pengiriman temuan ke Security Hub CSPM, Anda dapat menggunakan konsol CSPM Security Hub atau API.

Lihat Menonaktifkan dan mengaktifkan alur temuan dari integrasi (konsol) atau Menonaktifkan alur temuan dari integrasi (Security Hub API, AWS CLI) di Panduan Pengguna.AWS Security Hub

Mencatat panggilan API IAM Access Analyzer dengan AWS CloudTrail

IAM Access Analyzer terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di IAM Access Analyzer. CloudTrail menangkap semua panggilan API untuk IAM Access Analyzer sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari konsol IAM Access Analyzer dan panggilan kode ke operasi API IAM Access Analyzer.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail peristiwa secara terus menerus ke bucket Amazon S3, termasuk peristiwa untuk IAM Access Analyzer. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara.

Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk IAM Access Analyzer, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Informasi IAM Access Analyzer di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi di IAM Access Analyzer, aktivitas tersebut direkam dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh acara terbaru di AWS akun Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk IAM Access Analyzer, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua AWS Wilayah. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file

log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran Umum untuk Membuat Jejak
- CloudTrail Layanan dan Integrasi yang Didukung
- Mengkonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua tindakan IAM Access Analyzer dicatat oleh CloudTrail dan didokumentasikan dalam Referensi API IAM Access Analyzer. Misalnya, panggilan ke`CreateAnalyzer`, `CreateArchiveRule` dan `ListFindings` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan itu dibuat dengan kredenal pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Memahami entri file log IAM Access Analyzer

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateAnalyzer` operasi yang dilakukan oleh sesi peran yang dianggap bernama “14 Juni `Alice-tempcreds` 2021". Sesi peran dikeluarkan oleh peran bernama`admin-tempcreds`.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
```

```
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Kunci filter IAM Access Analyzer

Anda dapat menggunakan kunci filter di bawah ini untuk menentukan aturan arsip (`CreateArchiveRule`), memperbarui aturan arsip (`UpdateArchiveRule`), mengambil daftar temuan (`ListFindings`dan `ListFindingsV2`), atau mengambil daftar temuan pratinjau akses untuk sumber daya (`ListAccessPreviewFindings`). Tidak ada perbedaan antara menggunakan IAM API dan CloudFormation untuk mengonfigurasi aturan arsip.

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
sumber daya	Sumber Daya	ARN secara unik mengidentifikasi sumber daya yang dapat diakses oleh penanggung jawab eksternal. Untuk mempelajari selengkapnya, lihat nama sumber daya Amazon (ARNs).	String	✓Ya	✓Ya	✓Ya	Eksternal Internal Tidak terpakai
resourceT ype `AWS::S3 Bucket` | `AWS::IAI`	Jenis Sumber Daya	Jenis sumber daya yang dapat diakses oleh penanggung jawab eksternal.	String	✓Ya	✓Ya	✓Ya	Eksternal Internal Tidak terpakai

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
:Role | AWS::SQ :Queue | AWS::Lar da::Fun ion | AWS::Lar da::Lay Version |AWS::KM :Key | AWS::Se etsMana r::Secr | AWS::EF :FileSy em | AWS::EC :Snapsh | AWS::EC :Reposi ry | AWS::RD :DBSnap ot | AWS::RD :DBClus rSnapsh		**Note** Penganali sis akses internal tidak mendukung semua jenis sumber daya yang didukung oleh penganali sis akses eksternal. Penganali sis akses yang tidak digunakan hanya mendukung pengguna dan peran IAM. Untuk informasi selengkap nya, lihat IAM Access Analyzer mendukung jenis sumber					

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
| AWS::SN :Topic | AWS::S3 press::D rectoryB cket | AWS::Dyn moDB::Ta le | AWS::Dyn moDB::St eam | AWS::IAM :User		daya untuk akses eksternal dan internal.					
resourceC wnerAcco nt	Akun Pemilik Sumber Daya	12 digit ID AWS akun yang memiliki sumber daya. Untuk mempelaja ri lebih lanjut, lihat pengidentifikasi akun AWS.	Tali	✓Ya	✓Ya	✓Ya	Eksternal Internal Tidak terpakai
isPublic	Akses publik	Menunjukkan apakah temuan melaporkan sumber daya yang memiliki kebijakan yang memungkin kan akses publik.	Boolean	✓Ya	✓Ya	✓Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
FindingTy pe `Externa ccess` | `UnusedI Role` | `UnusedI UserAcc sKey` | `UnusedI UserPas ord` | `UnusedP mission` | `Interna ccess`	Jenis temuan	Jenis temuannya . Untuk penganali sis akses eksternal, tipenya adalah`ExternalA ccess` . Untuk penganalisis akses yang tidak digunakan , jenisnya dapat berupa`UnusedIAM Role` ,, `UnusedIAM UserAcces sKey` `UnusedIAM UserPassword` , atau. `UnusedPer mission` Untuk penganalisis akses internal, tipenya adalah`InternalA ccess` .	String	⊘Ya	⊘Ya	⊘Ya	Eksternal Internal Tidak terpakai

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
resourceC ontrolPol icyPemba asan `APPLIED	APPLICAI E	FAILED_T _EVALUA _RCP	NOT_APPI CABLE`	Pembatas n kebijakan kontrol sumber daya (RCP)	Jenis pembatasa n yang diterapka n oleh pemilik sumber daya dengan kebijakan kontrol sumber daya Organizat ions (RCP). Untuk informasi selengkapnya tentang nilai kunci filter ini, lihat ExternalA ccessDetailsdan InternalAccessDeta ilsdi Referensi API IAM Access Analyzer.	String	✓Ya	✓Ya	✓Ya	Eksternal Internal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
serviceCo ntrolPoli cyPembat san `APPLIED` | `APPLICAI E` | `FAILED_ _EVALUA _SCP` | `NOT_APPI CABLE`	Pembatas n kebijakan kontrol layanan (SCP)	Jenis pembatasa n yang diterapka n oleh kebijakan kontrol layanan Organizat ions (SCP). Untuk informasi selengkapnya tentang nilai kunci filter ini, lihat InternalAccessDeta ilsdi Referensi API IAM Access Analyzer.	String	⊘Ya	⊘Ya	⊘Ya	Internal
status `ACTIVE` | `ARCHIVEI` | `RESOLVEI`	Status	Status temuan saat ini.	String	⊗Tidak	⊘Ya	⊘Ya	Eksternal Internal Tidak terpakai
kesalahar	Kesalahar	Menunjukkan kesalahan yang dilaporkan untuk temuan.	String	⊘Ya	⊘Ya	⊘Ya	Eksternal Internal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
Principal .aws	AWS Akun	Akun tersebut memberikan akses ke sumber daya di `Principal` bidang temuan. Masukkan ID AWS akun 12 digit atau ARN dari pengguna atau peran AWS eksternal. Untuk mempelajari lebih lanjut, lihat pengidentifikasi akun AWS.	Tali	Ya	Ya	Ya	Eksternal
principal .federate d	Pengguna Federasi	ARN identitas gabungan yang memiliki akses ke sumber daya di temuan. Untuk mempelajari lebih lanjut, lihat Penyedia identitas dan federasi identitas	String	Ya	Ya	Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.aws: PrincipalArn	ARN Utama	ARN penanggung jawab (pengguna, peran, atau kelompok IAM) yang diindikasikan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	✓Ya	✓Ya	✓Ya	Eksternal
condition.aws: ID PrincipalOrg	Utama OrGid	Pengidentifikasi organisasi penganggung jawab ditunjukkan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	✓Ya	✓Ya	✓Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.aws: PrincipalOrgPaths	Kepala Sekolah OrgPaths	ID organisasi atau unit organisasi (OU) ditunjukkan sebagai kondisi untuk akses sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	Ya	Ya	Ya	Eksternal
kondisi.aws: SourceIp	Sumber IP	Alamat IP yang memungkinkan akses penanggung jawab ke sumber daya ketika menggunakan alamat IP yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	Alamat IP	Ya	Ya	Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.aws: SourceVp	Sumber VPC	ID VPC yang memungkinkan akses penanggung jawab ke sumber daya saat menggunakan VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	Ya	Ya	Ya	Eksternal
kondisi.aws: UserId	ID Pengguna	ID pengguna pengguna IAM dari akun eksternal yang ditunjukkan sebagai kondisi untuk akses ke sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	Ya	Ya	Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.aws: VpceAccount	Akun VPCE	ID akun dari titik akhir VPC yang memungkinkan akses utama ke sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	✓Ya	✓Ya	✓Ya	Eksternal Internal
kondisi.aws: SourceVpArn	Sumber VPC Arn	VPC ARN yang memungkinkan akses utama ke sumber daya saat menggunakan VPC yang ditentukan. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	ARN	✓Ya	✓Ya	✓Ya	Eksternal
condition.aws: IDVpceOrg	VPCE OrGid	ID organisasi untuk titik akhir VPC yang memungkinkan akses utama ke sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String	✓Ya	✓Ya	✓Ya	Eksternal Internal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.aws: VpceOrgPaths	VPCE OrgPaths	Unit organisasi (OU) untuk titik akhir VPC yang memungkinkan akses utama ke sumber daya. Untuk mempelajari lebih lanjut, lihat kunci konteks syarat global AWS.	String (daftar)	✓Ya	✓Ya	✓Ya	Eksternal Internal
condition.cognito-identity.amazonav.com:id	Penonton Cognito	ID pool identitas Amazon Cognito ditentukan sebagai kondisi untuk akses peran IAM di temuan. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	✓Ya	✓Ya	✓Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
condition .graph.fa cebook.cc m:app_id	ID Aplikasi Facebook	ID aplikasi Facebook (atau ID situs) yang ditentukan sebagai kondisi untuk memungkinkan Masuk dengan akses federasi Facebook ke peran IAM di temuan. Untuk mempelaja ri selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	⊘Ya	⊘Ya	⊘Ya	Eksternal
condition .accounts .google.c om:id	Pemirsa Google	ID aplikasi Google yang ditetapkan sebagai kondisi untuk mengakses peran IAM. Untuk mempelajari selengkapnya, lihat IAM dan AWS STS kunci-kunci konteks kondisi.	String	⊘Ya	⊘Ya	⊘Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
kondisi.kms: CallerAccount	ID Kunci KMS	ID AWS akun yang memiliki entitas pemanggil (pengguna IAM, peran atau pengguna root akun) yang digunakan oleh panggilan layanan. AWS KMS Untuk mempelajari lebih lanjut, lihat Kunci kondisi untuk AWS Key Management Service.	String	Ya	Ya	Ya	Eksternal
kondisi.www.amazo .com:app_id	ID Aplikasi Amazon	ID aplikasi Amazon (atau ID situs) yang ditentukan sebagai kondisi untuk memungkinkan akses federasi Login with Amazon ke peran. Untuk mempelajari lebih lanjut, lihat	String	Ya	Ya	Ya	Eksternal

Kriteria	Konsol Manajeme AWS lapangan	Deskripsi	Jenis	Aturan arsip	Daftar temuan	Daftar temuan pratinjau akses	Jenis penganali sis yang didukung
id	Menemuk ID	ID temuan.	String	⊗Tidak	⊘Ya	⊘Ya	Eksternal Internal Tidak terpakai
ChangeTy e `CHANGED | NEW | UNCHANGI`		Menyediakan konteks tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada yang diidentif ikasi dalam IAM Access Analyzer.	String	⊗Tidak	⊗Tidak	⊘Ya	Eksternal
existingF indingId		ID temuan yang ada di IAM Access Analyzer, disediaka n hanya untuk temuan yang ada di pratinjau akses.	String	⊗Tidak	⊗Tidak	⊘Ya	Eksternal
existingF indingSta tus		Status yang ada dari temuan, disediakan hanya untuk temuan yang ada di pratinjau akses.	String	⊗Tidak	⊗Tidak	⊘Ya	Eksternal

Menggunakan peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menggunakan peran terkait layanan IAM. Peran terkait layanan adalah jenis unik peran IAM yang ditautkan langsung ke IAM Access Analyzer. Peran terkait layanan telah ditentukan sebelumnya oleh IAM Access Analyzer dan mencakup semua izin yang diperlukan fitur untuk memanggil layanan lain atas nama Anda. AWS

Peran terkait layanan membuat pengaturan IAM Access Analyzer lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. IAM Access Analyzer mendefinisikan izin peran terkait layanannya, dan kecuali ditentukan lain, hanya IAM Access Analyzer yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Layanan AWS yang Berfungsi dengan IAM dan cari layanan yang memiliki Ya di kolom Peran Terkait Layanan. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer menggunakan peran terkait layanan bernama AWSServiceRoleForAccessAnalyzer— Izinkan Penganalisis Akses untuk menganalisis metadata sumber daya untuk akses eksternal dan menganalisis aktivitas guna mengidentifikasi akses yang tidak digunakan.

Peran AWSService RoleForAccessAnalyzer terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `access-analyzer.amazonaws.com`

Kebijakan izin peran bernama `AccessAnalyzerServiceRolePolicy`memungkinkan IAM Access Analyzer menyelesaikan tindakan pada sumber daya tertentu.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, silakan lihat Izin Peran Tertaut Layanan di Panduan Pengguna IAM.

Membuat peran terkait layanan untuk IAM Access Analyzer

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mengaktifkan Access Analyzer di Konsol Manajemen AWS atau AWS API, IAM Access Analyzer akan membuat peran terkait layanan untuk Anda. Peran terkait layanan yang sama digunakan di semua Wilayah tempat Anda mengaktifkan IAM Access Analyzer. Akses eksternal dan temuan akses yang tidak digunakan menggunakan peran terkait layanan yang sama.

> **Note**
>
> IAM Access Analyzer adalah Regional. Anda harus mengaktifkan IAM Access Analyzer di setiap Wilayah secara independen.

Jika Anda menghapus peran terkait layanan ini, IAM Access Analyzer akan membuat ulang peran tersebut saat Anda membuat penganalisis selanjutnya.

Anda juga dapat menggunakan konsol IAM untuk membuat peran yang terhubung dengan layanan dengan kasus penggunaan Penganalisis Akses. Di AWS CLI atau AWS API, buat peran terkait layanan dengan nama `access-analyzer.amazonaws.com` layanan. Untuk informasi lebih lanjut, lihat Membuat Peran yang Terhubung dengan Layanan di Panduan Pengguna IAM. Jika Anda menghapus peran tertaut layanan ini, Anda dapat mengulang proses yang sama untuk membuat peran tersebut lagi.

Mengedit peran terkait layanan untuk IAM Access Analyzer

IAM Access Analyzer tidak mengizinkan Anda mengedit peran terkait AWSService RoleForAccessAnalyzer layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin merujuk peran tersebut. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit Peran Tertaut Layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk IAM Access Analyzer

Jika Anda tidak lagi memerlukan penggunaan fitur atau layanan yang memerlukan peran terkait layanan, kami menyarankan Anda untuk menghapus peran tersebut. Dengan begitu Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran terkait layanan sebelum menghapusnya secara manual.

Jika IAM Access Analyzer diaktifkan di satu atau beberapa wilayah di Anda AWS Organizations, Anda harus menghapus semua penganalisis di semua wilayah untuk organisasi Anda sebelum mencoba menghapus peran ini.

> **Note**
>
> Jika IAM Access Analyzer menggunakan peran saat Anda mencoba menghapus sumber daya, maka penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus sumber daya IAM Access Analyzer yang digunakan oleh peran AWSService RoleForAccessAnalyzer

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bagian Laporan akses, di bawah Penganalisis akses, pilih Penganalisis.
3. Pilih penganalisis tempat Anda ingin menghapus sumber daya IAM Access Analyzer yang dilampirkan ke peran terkait layanan.
4. Pilih Hapus.
5. Untuk mengonfirmasi bahwa Anda ingin menghapus penganalisis, masukkan **`delete`**, lalu pilih Hapus.

Untuk menghapus peran tertaut layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran AWSService RoleForAccessAnalyzer terkait layanan. Untuk informasi selengkapnya, silakan lihat Menghapus Peran Terkait Layanan di Panduan Pengguna IAM.

Wilayah yang Didukung untuk peran terkait layanan IAM Access Analyzer

IAM Access Analyzer mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Wilayah dan Titik Akhir AWS.

Akses pratinjau

Selain membantu Anda mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal, AWS IAM Access Analyzer juga menampilkan pratinjau temuan IAM Access Analyzer sebelum menerapkan izin sumber daya sehingga Anda dapat memvalidasi bahwa perubahan kebijakan hanya

memberikan akses publik dan lintas akun yang dimaksudkan ke sumber daya Anda. Ini membantu Anda memulai dengan akses eksternal yang dimaksudkan ke sumber daya Anda.

Anda dapat melihat pratinjau dan memvalidasi akses publik dan lintas akun ke bucket Amazon S3 Anda di konsol Amazon S3. Anda juga dapat menggunakan API Penganalisis Akses IAM untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, AWS KMS kunci, peran IAM, antrian Amazon SQS, dan rahasia Manajer Rahasia dengan memberikan izin yang diusulkan untuk sumber daya Anda.

Topik

- Mempratinjau akses di konsol Amazon S3
- Mempratinjau akses dengan IAM Access Analyzer APIs

Mempratinjau akses di konsol Amazon S3

Setelah menyelesaikan kebijakan bucket di konsol Amazon S3, Anda memiliki opsi untuk melihat pratinjau akses publik dan lintas akun ke bucket Amazon S3 Anda. Anda dapat memvalidasi bahwa perubahan kebijakan hanya memberikan akses eksternal yang diinginkan sebelum Anda memilih Simpan perubahan. Langkah opsional ini memungkinkan Anda untuk melihat pratinjau temuan AWS Identity and Access Management Access Analyzer untuk bucket Anda. Anda dapat memvalidasi apakah perubahan kebijakan memperkenalkan temuan baru atau menyelesaikan temuan yang ada untuk akses eksternal. Anda dapat melewati langkah validasi ini dan menyimpan kebijakan bucket Amazon S3 kapan saja.

Untuk melakukan pratinjau akses eksternal ke bucket, Anda harus memiliki penganalisis akun aktif di wilayah bucket dengan akun sebagai zona kepercayaan. Anda juga harus memiliki izin yang diperlukan untuk menggunakan IAM Access Analyzer dan akses pratinjau. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat. Memulai dengan AWS Identity and Access Management Access Analyzer

Untuk melihat pratinjau akses ke bucket Amazon S3 saat Anda membuat atau mengedit kebijakan bucket

1. Setelah selesai membuat atau mengedit kebijakan bucket, pastikan kebijakan Anda adalah kebijakan bucket Amazon S3 yang valid. Kebijakan ARN harus sesuai dengan bucket ARN dan elemen kebijakan harus valid.
2. Di bawah kebijakan, di bagian Akses eksternal pratinjau, pilih penganalisis akun aktif, lalu pilih Pratinjau. Pratinjau temuan IAM Access Analyzer dihasilkan untuk bucket Anda. Pratinjau

menganalisis kebijakan bucket Amazon S3 yang ditampilkan, bersama dengan izin bucket yang ada. Ini termasuk pengaturan BPA bucket dan akun, bucket ACL, jalur akses Amazon S3, dan titik akses multi-wilayah yang terpasang pada bucket, serta kebijakan serta pengaturan BPA mereka.

3. Ketika pratinjau akses selesai, pratinjau temuan IAM Access Analyzer ditampilkan. Setiap temuan melaporkan instance prinsipal di luar akun dengan akses ke bucket Anda setelah Anda menyimpan polis. Anda dapat memvalidasi akses ke bucket Anda dengan meninjau setiap temuan. Header temuan menyediakan ringkasan akses dan Anda dapat memperluas temuan untuk meninjau detail temuan. Badge temuan memberikan konteks tentang bagaimana menyimpan kebijakan bucket akan mengubah akses ke bucket. Misalnya, mereka membantu Anda memvalidasi apakah perubahan kebijakan memperkenalkan temuan baru atau menyelesaikan temuan yang ada untuk akses eksternal:

   a. Baru — menunjukkan temuan untuk akses eksternal baru yang akan diperkenalkan oleh kebijakan tersebut.

   b. Terselesaikan — menunjukkan temuan untuk akses eksternal yang sudah ada yang akan dihapus oleh kebijakan tersebut.

   c. Diarsipkan — menunjukkan temuan untuk akses eksternal baru yang akan diarsipkan secara otomatis, berdasarkan aturan arsip untuk penganalisis yang menentukan kapan temuan harus ditandai sebagaimana dimaksud.

   d. Ada — menunjukkan temuan yang ada untuk akses eksternal yang akan tetap tidak berubah.

   e. Publik — jika sebuah temuan adalah untuk akses publik ke sumber daya, itu akan memiliki lencana Publik, di samping salah satu lencana di atas.

4. Jika Anda mengidentifikasi akses eksternal yang tidak ingin Anda perkenalkan atau hapus, Anda dapat merevisi kebijakan dan kemudian memilih Pratinjau lagi sampai Anda telah mencapai akses eksternal yang Anda inginkan. Jika Anda memiliki temuan berlabel Publik, kami sarankan Anda merevisi kebijakan untuk menghapus akses publik sebelum Anda memilih Simpan perubahan. Pratinjau akses adalah langkah opsional dan Anda dapat memilih Simpan perubahan kapan saja.

Mempratinjau akses dengan IAM Access Analyzer APIs

Anda dapat menggunakan IAM Access Analyzer APIs untuk melihat pratinjau akses publik dan lintas akun untuk bucket Amazon S3, AWS KMS kunci, peran IAM, antrian Amazon SQS, dan rahasia

Secrets Manager. Anda dapat melihat pratinjau akses dengan memberikan izin yang diajukan untuk sumber daya yang ada yang Anda miliki atau sumber daya baru yang ingin Anda deploy.

Untuk melihat akses eksternal ke sumber daya Anda, Anda harus memiliki penganalisis akun aktif untuk akun dan wilayah sumber daya. Anda juga harus memiliki izin yang diperlukan untuk menggunakan IAM Access Analyzer dan akses pratinjau. Untuk informasi selengkapnya tentang mengaktifkan IAM Access Analyzer dan izin yang diperlukan, lihat. Memulai dengan AWS Identity and Access Management Access Analyzer

Untuk melihat pratinjau akses untuk sumber daya, Anda dapat menggunakan operasi `CreateAccessPreview` dan menyediakan penganalisis ARN dan konfigurasi kontrol akses untuk sumber daya. Layanan mengembalikan ID unik untuk pratinjau akses, yang dapat Anda gunakan untuk memeriksa status pratinjau akses dengan operasi `GetAccessPreview`. Ketika statusnya adalah `Completed`, Anda dapat menggunakan operasi `ListAccessPreviewFindings` untuk mengambil temuan yang dihasilkan untuk pratinjau akses. Operasi `GetAccessPreview` dan `ListAccessPreviewFindings` akan mengambil pratinjau akses dan temuan yang dibuat dalam waktu sekitar 24 jam.

Setiap temuan yang diambil berisi detail temuan yang menggambarkan akses. Status pratinjau dari temuan yang menjelaskan apakah temuan akan berupa `Active`, `Archived`, atau `Resolved` setelah deployment izin, dan `changeType`. Konteks `changeType` menyediakan tentang bagaimana temuan pratinjau akses dibandingkan dengan akses yang ada yang diidentifikasi dalam IAM Access Analyzer:

- Baru — temuannya adalah untuk akses yang baru diperkenalkan.
- Tidak berubah — temuan pratinjau adalah temuan yang ada yang akan tetap tidak berubah.
- Berubah - temuan pratinjau adalah temuan yang ada dengan perubahan status.

`status` dan `changeType` membantu Anda memahami bagaimana konfigurasi sumber daya akan mengubah akses sumber daya yang ada. Jika `changeType` ada `Unchanged` atau Diubah, temuan juga akan berisi ID dan status temuan yang ada di IAM Access Analyzer. Misalnya, temuan `Changed` dengan status pratinjau `Resolved` dan status saat ini `Active` mengindikasikan temuan `Active` yang ada untuk sumber daya akan menjadi `Resolved` sebagai hasil dari perubahan izin yang diajukan.

Anda dapat menggunakan operasi `ListAccessPreviews` untuk memperoleh daftar pratinjau akses untuk penganalisis tertentu. Operasi ini akan mengambil informasi tentang pratinjau akses yang dibuat dalam waktu sekitar satu jam.

Secara umum, jika pratinjau akses adalah untuk sumber daya yang ada dan Anda meninggalkan opsi konfigurasi yang tidak ditentukan, pratinjau akses akan menggunakan konfigurasi sumber daya yang ada secara default. Jika pratinjau akses adalah untuk sumber daya baru dan Anda meninggalkan opsi konfigurasi yang tidak ditentukan, pratinjau akses akan menggunakan nilai default tergantung jenis sumber daya. Untuk kasus konfigurasi untuk setiap jenis sumber daya, lihat di bawah ini.

Pratinjau akses ke bucket Amazon S3 Anda

Untuk membuat pratinjau akses untuk bucket Amazon S3 baru atau bucket Amazon S3 yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi bucket dengan menentukan kebijakan bucket Amazon S3, bucket, pengaturan bucket BPA ACLs, dan jalur akses Amazon S3, termasuk titik akses multi-wilayah, yang dilampirkan ke bucket.

> **Note**
>
> Sebelum mencoba membuat pratinjau akses untuk bucket baru, kami sarankan Anda memanggil operasi Amazon HeadBucketS3 untuk memeriksa apakah bucket bernama sudah ada. Operasi ini berguna untuk menentukan apakah bucket ada dan Anda memiliki izin untuk mengaksesnya.

Kebijakan Bucket — Jika konfigurasi untuk bucket Amazon S3 yang sudah ada dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses menggunakan kebijakan yang ada yang dilampirkan ke bucket. Jika pratinjau akses untuk sumber daya baru dan Anda tidak menentukan kebijakan bucket Amazon S3, pratinjau akses akan mengasumsikan bucket tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan bucket yang ada, Anda dapat menentukan string kosong. Untuk informasi selengkapnya tentang kebijakan bucket yang didukung, lihat Contoh kebijakan bucket.

Hibah ACL Bucket — Anda dapat mengusulkan hingga 100 hibah ACL per ember. Jika konfigurasi izin diusulkan untuk bucket yang ada, pratinjau akses menggunakan daftar diusulkan konfigurasi izin di tempat izin yang ada. Jika tidak, pratinjau akses menggunakan izin yang ada untuk bucket.

Titik akses bucket — Analisis ini mendukung hingga 100 titik akses, termasuk titik akses multi-wilayah, per ember, termasuk hingga sepuluh titik akses baru yang dapat Anda usulkan per ember. Jika konfigurasi titik akses Amazon S3 yang diusulkan adalah untuk bucket yang ada, pratinjau akses menggunakan konfigurasi titik akses yang diusulkan sebagai pengganti titik akses yang ada. Untuk mengusulkan jalur akses tanpa kebijakan, Anda dapat memberikan string kosong sebagai kebijakan

titik akses. Untuk informasi selengkapnya tentang batas kebijakan titik akses, lihat Pembatasan dan batasan titik akses.

Blokir konfigurasi akses publik — Jika konfigurasi yang diusulkan adalah untuk bucket Amazon S3 yang ada dan Anda tidak menentukan konfigurasi, pratinjau akses menggunakan setelan yang ada. Jika konfigurasi yang diusulkan adalah untuk bucket baru dan Anda tidak menentukan konfigurasi bucket BPA, pratinjau akses akan digunakan`false`. Jika konfigurasi yang diusulkan adalah untuk titik akses baru atau titik akses multi-wilayah, dan Anda tidak menentukan konfigurasi BPA titik akses, pratinjau akses menggunakan. `true`

Pratinjau akses ke AWS KMS kunci Anda

Untuk membuat pratinjau akses untuk AWS KMS kunci baru atau AWS KMS kunci yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi AWS KMS kunci dengan menentukan kebijakan kunci dan konfigurasi AWS KMS hibah.

AWS KMS kebijakan kunci — Jika konfigurasi untuk kunci yang ada dan Anda tidak menentukan kebijakan kunci, pratinjau akses menggunakan kebijakan yang ada untuk kunci tersebut. Jika pratinjau akses adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan kunci, maka pratinjau akses menggunakan kebijakan kunci default. Kebijakan kunci yang diusulkan tidak boleh string kosong.

AWS KMS hibah — Analisis ini mendukung hibah hingga 100 KMS per konfigurasi*.* Jika konfigurasi hibah yang diusulkan adalah untuk kunci yang ada, pratinjau akses menggunakan daftar konfigurasi hibah yang diusulkan sebagai pengganti hibah yang ada. Jika tidak, pratinjau akses menggunakan izin yang ada untuk kunci.

Pratinjau akses ke peran IAM Anda

Untuk membuat pratinjau akses untuk peran IAM baru atau peran IAM yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi peran IAM dengan menentukan kebijakan kepercayaan.

Kebijakan kepercayaan peran — Jika konfigurasi untuk peran IAM baru, Anda harus menentukan kebijakan kepercayaan. Jika konfigurasi untuk peran IAM yang sudah ada yang Anda miliki dan Anda tidak mengusulkan kebijakan kepercayaan, pratinjau akses menggunakan kebijakan kepercayaan yang ada untuk peran tersebut. Kebijakan kepercayaan yang diusulkan tidak boleh string kosong.

Pratinjau akses ke antrean Amazon SQS

Untuk membuat pratinjau akses untuk antrean Amazon SQS baru atau antrean Amazon SQS yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi antrian Amazon SQS dengan menentukan kebijakan Amazon SQS untuk antrean.

Kebijakan antrian Amazon SQS — Jika konfigurasi untuk antrean Amazon SQS yang ada dan Anda tidak menentukan kebijakan Amazon SQS, pratinjau akses menggunakan kebijakan Amazon SQS yang ada untuk antrean. Jika pratinjau akses untuk sumber daya baru dan Anda tidak menentukan kebijakan, pratinjau akses akan mengasumsikan antrean Amazon SQS tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan antrian Amazon SQS yang ada, Anda dapat menentukan string kosong untuk kebijakan Amazon SQS.

Pratinjau akses ke rahasia Secrets Manager

Untuk membuat pratinjau akses untuk rahasia Secrets Manager baru atau rahasia Secrets Manager yang sudah ada yang Anda miliki, Anda dapat mengusulkan konfigurasi rahasia Secrets Manager dengan menentukan kebijakan rahasia dan kunci AWS KMS enkripsi opsional.

Kebijakan rahasia — Jika konfigurasi untuk rahasia yang ada dan Anda tidak menentukan kebijakan rahasia, pratinjau akses menggunakan kebijakan yang ada untuk rahasia tersebut. Jika pratinjau akses adalah untuk sumber daya baru dan Anda tidak menentukan kebijakan, pratinjau akses mengasumsikan rahasia tanpa kebijakan. Untuk mengusulkan penghapusan kebijakan yang ada, Anda dapat menentukan string kosong.

AWS KMS kunci enkripsi — Jika konfigurasi yang diusulkan adalah untuk rahasia baru dan Anda tidak menentukan ID AWS KMS kunci, pratinjau akses menggunakan kunci KMS default AWS akun. Jika Anda menentukan string kosong untuk ID AWS KMS kunci, pratinjau akses menggunakan kunci KMS default AWS akun.

Memeriksa untuk memvalidasi kebijakan

IAM Access Analyzer menyediakan pemeriksaan kebijakan yang membantu memvalidasi kebijakan IAM Anda sebelum Anda melampirkannya ke entitas. Ini termasuk pemeriksaan kebijakan dasar yang disediakan oleh validasi kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan dan praktik AWS terbaik. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda.

Anda dapat menggunakan pemeriksaan kebijakan khusus untuk memeriksa akses baru berdasarkan standar keamanan Anda. Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail

selengkapnya tentang harga, lihat harga https://aws.amazon.com/iam/access-analyzer/pricing IAM Access Analyzer.

Cara kerja pemeriksaan kebijakan khusus

Anda dapat memvalidasi kebijakan terhadap standar keamanan yang ditentukan menggunakan pemeriksaan kebijakan AWS Identity and Access Management Access Analyzer khusus. Anda dapat menjalankan jenis pemeriksaan kebijakan kustom berikut ini:

- Periksa terhadap kebijakan referensi: Saat mengedit kebijakan, Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan kebijakan referensi, seperti versi kebijakan yang sudah ada. Anda dapat menjalankan pemeriksaan ini saat mengedit kebijakan menggunakan AWS Command Line Interface (AWS CLI), IAM Access Analyzer API (API), atau editor kebijakan JSON di konsol IAM.

  > **Note**
  >
  > Pemeriksaan kebijakan kustom IAM Access Analyzer mengizinkan wildcard dalam `Principal` elemen untuk kebijakan sumber daya referensi.

- Periksa terhadap daftar tindakan atau sumber daya IAM: Anda dapat memeriksa untuk memastikan bahwa tindakan atau sumber daya IAM tertentu tidak diizinkan oleh kebijakan Anda. Jika hanya tindakan yang ditentukan, IAM Access Analyzer memeriksa akses tindakan pada semua sumber daya dalam kebijakan. Jika hanya sumber daya yang ditentukan, maka IAM Access Analyzer memeriksa tindakan mana yang memiliki akses ke sumber daya yang ditentukan. Jika tindakan dan sumber daya ditentukan, maka Penganalisis Akses IAM memeriksa tindakan yang ditentukan mana yang memiliki akses ke sumber daya yang ditentukan. Anda dapat menjalankan pemeriksaan ini saat membuat atau mengedit kebijakan menggunakan API AWS CLI atau.
- Periksa akses publik: Anda dapat memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu. Anda dapat menjalankan pemeriksaan ini saat membuat atau mengedit kebijakan menggunakan API AWS CLI atau. Jenis pemeriksaan kebijakan kustom ini berbeda dengan pratinjau akses karena pemeriksaan tidak memerlukan akun atau konteks penganalisis akses eksternal. Pratinjau akses memungkinkan Anda untuk melihat pratinjau temuan Penganalisis Akses IAM sebelum menerapkan izin sumber daya, sementara pemeriksaan khusus menentukan apakah akses publik dapat diberikan oleh kebijakan.

Biaya dikaitkan dengan setiap pemeriksaan kebijakan kustom. Untuk detail selengkapnya tentang harga, lihat harga https://aws.amazon.com/iam/access-analyzer/pricing IAM Access Analyzer.

Anda dapat menjalankan pemeriksaan kebijakan khusus pada kebijakan berbasis identitas dan sumber daya. Pemeriksaan kebijakan khusus tidak bergantung pada teknik pencocokan pola atau memeriksa log akses untuk menentukan apakah akses baru atau akses tertentu diizinkan oleh kebijakan. Mirip dengan temuan akses eksternal, pemeriksaan kebijakan khusus dibangun di Zelkova. Zelkova menerjemahkan kebijakan IAM menjadi pernyataan logis yang setara, dan menjalankan serangkaian pemecah logis tujuan umum dan khusus (teori modulo kepuasan) terhadap masalah tersebut. Untuk memeriksa akses baru atau tertentu, Penganalisis Akses IAM menerapkan Zelkova berulang kali ke kebijakan. Kueri menjadi semakin spesifik untuk mengkarakterisasi kelas perilaku yang diizinkan kebijakan berdasarkan isi kebijakan. Untuk informasi lebih lanjut tentang teori modulo kepuasan, lihat Teori Modulo Ke puasan.

Dalam kasus yang jarang terjadi, IAM Access Analyzer tidak dapat sepenuhnya menentukan apakah pernyataan kebijakan memberikan akses baru atau tertentu. Dalam kasus tersebut, kesalahan terjadi di sisi menyatakan positif palsu dengan gagal dalam pemeriksaan kebijakan khusus. IAM Access Analyzer dirancang untuk memberikan evaluasi kebijakan yang komprehensif dan berusaha untuk meminimalkan negatif palsu. Pendekatan ini berarti bahwa IAM Access Analyzer memberikan jaminan tingkat tinggi bahwa pemeriksaan yang lulus berarti akses tidak diberikan oleh kebijakan. Anda dapat memeriksa pemeriksaan yang gagal secara manual dengan meninjau pernyataan kebijakan yang dilaporkan dalam respons dari IAM Access Analyzer.

Contoh kebijakan referensi untuk memeriksa akses baru

Anda dapat menemukan contoh kebijakan referensi dan mempelajari cara menyiapkan dan menjalankan pemeriksaan kebijakan kustom untuk akses baru di repositori sampel pemeriksaan kebijakan kustom IAM Access Analyzer. GitHub

Sebelum menggunakan contoh-contoh ini

Sebelum Anda menggunakan kebijakan referensi sampel ini, lakukan hal berikut:

- Tinjau dan sesuaikan kebijakan referensi dengan cermat untuk kebutuhan unik Anda.
- Uji secara menyeluruh kebijakan referensi di lingkungan Anda dengan Layanan AWS yang Anda gunakan.

  Kebijakan referensi menunjukkan implementasi dan penggunaan pemeriksaan kebijakan kustom. Contoh-contoh tersebut tidak dimaksudkan untuk ditafsirkan sebagai rekomendasi

AWS resmi atau praktik terbaik untuk diimplementasikan persis seperti yang ditunjukkan. Merupakan tanggung jawab Anda untuk secara hati-hati menguji kebijakan referensi untuk kesesuaiannya untuk menyelesaikan persyaratan keamanan untuk lingkungan Anda.

- Pemeriksaan kebijakan khusus bersifat agnostik terhadap lingkungan dalam analisisnya. Analisis mereka hanya mempertimbangkan informasi yang terkandung dalam kebijakan input. Misalnya, pemeriksaan kebijakan kustom tidak dapat memeriksa apakah akun adalah anggota AWS organisasi tertentu. Oleh karena itu, pemeriksaan kebijakan kustom tidak dapat membandingkan akses baru berdasarkan nilai kunci kondisi untuk kunci `aws:PrincipalOrgId` dan `aws:PrincipalAccount` kondisi.

Memeriksa pemeriksaan kebijakan kustom yang gagal

Ketika pemeriksaan kebijakan kustom gagal, respons dari IAM Access Analyzer menyertakan ID pernyataan (`Sid`) pernyataan pernyataan yang menyebabkan pemeriksaan gagal. Meskipun ID pernyataan adalah elemen kebijakan opsional, sebaiknya tambahkan ID pernyataan untuk setiap pernyataan kebijakan. Pemeriksaan kebijakan kustom juga mengembalikan indeks pernyataan untuk membantu mengidentifikasi alasan kegagalan pemeriksaan. Indeks pernyataan mengikuti penomoran berbasis nol, di mana pernyataan pertama dirujuk sebagai 0. Ketika ada beberapa pernyataan yang menyebabkan pemeriksaan gagal, cek hanya mengembalikan satu ID pernyataan pada satu waktu. Kami menyarankan Anda memperbaiki pernyataan yang disorot dalam alasan dan menjalankan kembali pemeriksaan sampai lulus.

Validasi kebijakan dengan IAM Access Analyzer

Anda dapat memvalidasi kebijakan Anda menggunakan validasi AWS Identity and Access Management Access Analyzer kebijakan. Anda dapat membuat atau mengedit kebijakan menggunakan editor kebijakan AWS CLI, AWS API, atau JSON di konsol IAM. IAM Access Analyzer memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik.AWS Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Untuk melihat daftar pemeriksaan kebijakan dasar yang dijalankan oleh IAM Access Analyzer, lihat. Referensi pemeriksaan validasi kebijakan IAM

Memvalidasi kebijakan di IAM (konsol)

Anda dapat melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer saat membuat atau mengedit kebijakan terkelola di konsol IAM. Anda juga dapat melihat temuan ini untuk pengguna inline atau kebijakan peran. IAM Access Analyzer tidak menghasilkan temuan ini untuk kebijakan grup sebaris.

Untuk melihat temuan yang dihasilkan oleh pemeriksaan kebijakan untuk kebijakan IAM JSON

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Mulai membuat atau mengedit kebijakan dengan salah satu metode berikut:
   a. Untuk membuat kebijakan terkelola baru, buka halaman Kebijakan buat kebijakan baru. Untuk informasi selengkapnya, lihat Membuat kebijakan menggunakan editor JSON.
   b. Untuk melihat pemeriksaan kebijakan untuk kebijakan terkelola pelanggan yang sudah ada, buka halaman Kebijakan, pilih nama kebijakan, lalu pilih Edit. Untuk informasi selengkapnya, lihat Menyunting kebijakan terkelola pelanggan (konsol).
   c. Untuk melihat pemeriksaan kebijakan untuk kebijakan sebaris pada pengguna atau peran, buka halaman Pengguna atau Peran, pilih nama pengguna atau peran, pilih nama kebijakan di tab Izin, lalu pilih Edit. Untuk informasi selengkapnya, lihat Mengedit kebijakan inline (konsol).
3. Di editor kebijakan, pilih tab JSON.
4. Di panel validasi kebijakan di bawah kebijakan, pilih salah satu atau beberapa tab berikut. Nama tab juga menunjukkan jumlah setiap jenis temuan untuk kebijakan Anda.
   - Keamanan — Lihat peringatan jika kebijakan Anda mengizinkan akses yang AWS mempertimbangkan risiko keamanan karena aksesnya terlalu permisif.
   - Kesalahan — Lihat kesalahan jika kebijakan Anda menyertakan baris yang mencegah kebijakan berfungsi.
   - Peringatan — Lihat peringatan jika kebijakan Anda tidak sesuai dengan praktik terbaik, tetapi masalahnya bukan risiko keamanan.
   - Saran — Lihat saran jika AWS merekomendasikan perbaikan yang tidak memengaruhi izin kebijakan.
5. Tinjau detail temuan yang disediakan oleh pemeriksaan kebijakan IAM Access Analyzer. Setiap temuan menunjukkan lokasi masalah yang dilaporkan. Untuk mempelajari lebih lanjut tentang

penyebab masalah dan cara mengatasinya, pilih tautan Pelajari selengkapnya di sebelah temuan. Anda juga dapat mencari pemeriksaan kebijakan yang terkait dengan setiap temuan di halaman referensi Pemeriksaan kebijakan Penganalisis Akses.

6. Tidak wajib. Jika Anda mengedit kebijakan yang ada, Anda dapat menjalankan pemeriksaan kebijakan khusus untuk menentukan apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada. Di panel validasi kebijakan di bawah kebijakan, pilih tab Periksa akses baru, lalu pilih Periksa kebijakan. Jika izin yang dimodifikasi memberikan akses baru, pernyataan akan disorot di panel validasi kebijakan. Jika Anda tidak bermaksud memberikan akses baru, perbarui pernyataan kebijakan dan pilih Periksa kebijakan hingga tidak ada akses baru yang terdeteksi. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer.

   > **Note**
   >
   > Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail lebih lanjut tentang harga, lihat harga IAM Access Analyzer.

7. Perbarui kebijakan Anda untuk menyelesaikan temuan.

   > **Important**
   >
   > Uji kebijakan baru atau diedit secara menyeluruh sebelum mengimplementasikannya dalam alur kerja produksi Anda.

8. Setelah selesai, pilih Selanjutnya. Validator Kebijakan melaporkan kesalahan sintaks yang tidak dilaporkan oleh IAM Access Analyzer.

   > **Note**
   >
   > Anda dapat beralih antara tab Visual dan JSON kapan saja. Namun, jika Anda membuat perubahan atau memilih Berikutnya di tab Visual, IAM mungkin merestrukturisasi kebijakan Anda untuk mengoptimalkannya untuk editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

9. Untuk kebijakan baru, pada halaman Tinjau dan buat, masukkan nama Kebijakan dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda. Kemudian pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

Untuk kebijakan yang ada, pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda. Pilih Setel versi baru ini sebagai default. kotak centang untuk menyimpan versi yang diperbarui sebagai versi default kebijakan. Kemudian pilih Simpan perubahan untuk menyimpan pekerjaan Anda.

Memvalidasi kebijakan menggunakan IAM Access Analyzer (AWS CLI atau API) AWS

Anda dapat melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer dari (). AWS Command Line Interface AWS CLI

Untuk melihat temuan yang dihasilkan oleh validasi kebijakan IAM Access Analyzer (AWS CLI atau API) AWS

Gunakan salah satu langkah berikut:

- AWS CLI: aws accessanalyzer validate-policy
- AWS API: ValidatePolicy

Referensi pemeriksaan validasi kebijakan IAM

Anda dapat memvalidasi kebijakan menggunakan validasi AWS Identity and Access Management Access Analyzer kebijakan. Anda dapat membuat atau mengedit kebijakan menggunakan AWS CLI, AWS API, atau editor kebijakan JSON di konsol IAM. IAM Access Analyzer memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan AWS praktik terbaik. Anda dapat melihat temuan pemeriksaan validasi kebijakan yang mencakup peringatan keamanan, kesalahan, peringatan umum, dan saran untuk kebijakan Anda. Temuan ini memberikan rekomendasi yang dapat ditindaklanjuti yang membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Daftar pemeriksaan kebijakan dasar yang disediakan oleh IAM Access Analyzer dibagikan di bawah ini. Tidak ada biaya tambahan yang terkait dengan menjalankan pemeriksaan validasi kebijakan. Untuk mempelajari lebih lanjut tentang memvalidasi kebijakan menggunakan validasi kebijakan, lihatValidasi kebijakan dengan IAM Access Analyzer.

Kesalahan — Akun ARN tidak diizinkan

Kode masalah: ARN_ACCOUNT_NOT_ALLOWED

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Menyelesaikan kesalahan

Hapus ID akun dari sumber daya ARN. ARN sumber daya untuk beberapa AWS layanan tidak mendukung penentuan ID akun.

Misalnya, Amazon S3 tidak mendukung ID akun sebagai namespace di bucket ARN. Nama bucket Amazon S3 unik secara global, dan namespace dibagikan oleh semua AWS akun. Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Istilah terkait

- Sumber daya kebijakan
- Pengidentifikasi Akun
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Wilayah ARN tidak diizinkan

Kode masalah: ARN_REGION_NOT_ALLOWED

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Menyelesaikan kesalahan

Hapus Wilayah dari sumber daya ARN. ARN sumber daya untuk beberapa AWS layanan tidak mendukung penentuan Wilayah.

Misalnya, IAM adalah layanan global. Bagian Wilayah dari ARN sumber daya IAM selalu dibiarkan kosong. Sumber daya IAM bersifat global, seperti AWS akun saat ini. Misalnya, setelah Anda masuk sebagai pengguna IAM, Anda dapat mengakses AWS layanan di wilayah geografis mana pun.

- Sumber daya kebijakan
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan - Ketidakcocokan tipe data

Kode masalah: DATA_TYPE_MISMATCH

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Menyelesaikan kesalahan

Perbarui teks untuk menggunakan tipe data yang didukung.

Misalnya, kunci kondisi `Version` global memerlukan tipe `String` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Kunci kondisi global
- Elemen kebijakan JSON IAM: Operator kondisi

Kesalahan - Duplikat kunci dengan kasus yang berbeda

Kode masalah: DUPLICATE_KEYS_WITH_DIFFERENT_CASE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Menyelesaikan kesalahan

Tinjau kunci kondisi serupa dalam blok kondisi yang sama dan gunakan kapitalisasi yang sama untuk semua instance.

Blok kondisi adalah teks dalam `Condition` elemen pernyataan kebijakan. Nama kunci kondisi tidak peka dengan huruf besar-kecil. Sensitivitas huruf besar/kecil dari nilai kunci kondisi tergantung pada

operator kondisi yang Anda gunakan. Untuk informasi selengkapnya tentang sensitivitas huruf besar/kecil di kunci kondisi, lihatElemen kebijakan IAM JSON: Kondisi.

Istilah terkait

- Kondisi
- Blok kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan - Tindakan tidak valid

Kode masalah: INVALID_ACTION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Menyelesaikan kesalahan

Tindakan yang Anda tentukan tidak valid. Hal ini dapat terjadi jika Anda salah mengetik awalan layanan atau nama tindakan. Untuk beberapa masalah umum, pemeriksaan kebijakan mengembalikan tindakan yang disarankan.

Istilah terkait

- Tindakan kebijakan
- AWS tindakan layanan

AWS kebijakan yang dikelola dengan kesalahan ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan terkel AWS ola berikut menyertakan tindakan tidak valid dalam pernyataan kebijakan mereka. Tindakan yang tidak valid tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola, Anda AWS disarankan untuk menghapus tindakan yang tidak valid dari kebijakan Anda.

- AmazonEMRFullAccessPolicy_v2
- CloudWatchSyntheticsFullAccess

Kesalahan — Akun ARN tidak valid

Kode masalah: INVALID_ARN_ACCOUNT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

Menyelesaikan kesalahan

Perbarui ID akun di sumber daya ARN. ID akun adalah bilangan bulat 12 digit. Untuk mempelajari cara melihat ID akun Anda, lihat M enemukan ID AWS akun Anda.

Istilah terkait

- Sumber daya kebijakan

- Pengidentifikasi Akun
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Awalan ARN tidak valid

Kode masalah: INVALID_ARN_PREFIX

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Menyelesaikan kesalahan

AWS ARN sumber daya harus menyertakan `arn:` awalan yang diperlukan.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Wilayah ARN tidak valid

Kode masalah: INVALID_ARN_REGION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Menyelesaikan kesalahan

Jenis sumber daya tidak didukung di Wilayah yang ditentukan. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- Nama dan kode wilayah

Kesalahan — Sumber daya ARN tidak valid

Kode masalah: INVALID_ARN_RESOURCE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Menyelesaikan kesalahan

Sumber daya ARN harus sesuai dengan spesifikasi untuk jenis sumber daya yang diketahui. Untuk melihat format ARN yang diharapkan untuk layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat jenis sumber daya dan format ARN.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Kasus layanan ARN tidak valid

Kode masalah: INVALID_ARN_SERVICE_CASE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid ARN service case: Update the service name {{service}} in the resource ARN to
 use all lowercase letters.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Update the service name {{service}} in the resource ARN to use all
 lowercase letters."
```

Menyelesaikan kesalahan

Layanan dalam sumber daya ARN harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan cari awalannya di kalimat pertama.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Jenis data kondisi tidak valid

Kode masalah: INVALID_CONDITION_DATA_TYPE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Menyelesaikan kesalahan

Nilai dalam pasangan kunci-nilai kondisi harus sesuai dengan tipe data kunci kondisi dan operator kondisi. Untuk melihat tipe data kunci kondisi untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat kunci kondisi untuk layanan tersebut.

Misalnya, kunci kondisi `CurrentTime` global mendukung operator `Date` kondisi. Jika Anda memberikan string atau bilangan bulat untuk nilai dalam blok kondisi, tipe data tidak akan cocok.

Istilah terkait

- Kondisi
- Blok kondisi
- Elemen kebijakan JSON IAM: Operator kondisi
- Kunci kondisi global

- AWS kunci kondisi layanan

Kesalahan — Format kunci kondisi tidak valid

Kode masalah: INVALID_CONDITION_KEY_FORMAT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Menyelesaikan kesalahan

Kunci dalam pasangan kunci-nilai kondisi harus sesuai dengan spesifikasi untuk layanan. Untuk melihat kunci kondisi untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat kunci kondisi untuk layanan tersebut.

Istilah terkait

- Kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan — Kondisi beberapa Boolean tidak valid

Kode masalah: INVALID_CONDITION_MULTIPLE_BOOLEAN

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Menyelesaikan kesalahan

Kunci dalam pasangan kunci-nilai kondisi mengharapkan nilai Boolean tunggal. Bila Anda memberikan beberapa nilai Boolean, kecocokan kondisi mungkin tidak mengembalikan hasil yang Anda harapkan.

Untuk melihat kunci kondisi untuk layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat kunci kondisi untuk layanan tersebut.

- Kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan — Operator kondisi tidak valid

Kode masalah: INVALID_CONDITION_OPERATOR

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Menyelesaikan kesalahan

Perbarui kondisi untuk menggunakan operator kondisi yang didukung.

Istilah terkait

- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kondisi
- Ikhtisar kebijakan JSON

Kesalahan - Efek tidak valid

Kode masalah: INVALID_EFFECT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Menyelesaikan kesalahan

Perbarui `Effect` elemen untuk menggunakan efek yang valid. Nilai yang valid untuk `Effect` adalah **`Allow`** dan **`Deny`**.

Istilah terkait

- Elemen efek
- Ikhtisar kebijakan JSON

Kesalahan — Kunci kondisi global tidak valid

Kode masalah: INVALID_GLOBAL_CONDITION_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Menyelesaikan kesalahan

Perbarui kunci kondisi dalam pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi global yang didukung.

Kunci kondisi global adalah kunci kondisi dengan `aws:` awalan. AWS layanan dapat mendukung kunci kondisi global atau menyediakan kunci khusus layanan yang menyertakan awalan layanan mereka. Misalnya, kunci kondisi IAM mencakup prefiks `iam:`. Untuk informasi selengkapnya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan yang kuncinya ingin Anda lihat.

Istilah terkait

- Kunci kondisi global

Kesalahan - Partisi tidak valid

Kode masalah: INVALID_PARTITION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Menyelesaikan kesalahan

Perbarui sumber daya ARN untuk menyertakan partisi yang didukung. Jika Anda menyertakan partisi yang didukung, maka layanan atau sumber daya mungkin tidak mendukung partisi yang Anda sertakan.

Partisi adalah sekelompok AWS wilayah. Setiap AWS akun dicakup ke satu partisi. Di Wilayah Klasik, gunakan `aws` partisi. Di Wilayah China, gunakan`aws-cn`.

Istilah terkait

- Nama Sumber Daya Amazon (ARN) - Partisi

Kesalahan — Elemen kebijakan tidak valid

Kode masalah: INVALID_POLICY_ELEMENT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid policy element: The policy element {{element}} is not valid.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy element {{element}} is not valid."
```

Menyelesaikan kesalahan

Perbarui kebijakan agar hanya menyertakan elemen kebijakan JSON yang didukung.

Istilah terkait

- Elemen kebijakan JSON

Kesalahan — Format utama tidak valid

Kode masalah: INVALID_PRINCIPAL_FORMAT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Menyelesaikan kesalahan

Perbarui prinsip untuk menggunakan format pasangan kunci-nilai yang didukung.

Anda dapat menentukan prinsip dalam kebijakan berbasis sumber daya, tetapi tidak kebijakan berbasis identitas.

Misalnya, untuk menentukan akses untuk semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- Elemen kebijakan JSON: Principal
- Identity-based kebijakan dan kebijakan berbasis sumber daya

Kesalahan — Kunci utama tidak valid

Kode masalah: INVALID_PRINCIPAL_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Menyelesaikan kesalahan

Perbarui kunci dalam pasangan kunci-nilai utama untuk menggunakan kunci utama yang didukung. Berikut ini adalah kunci utama yang didukung:

- AWS
- CanonicalUser
- Terfederasi
- Layanan

Istilah terkait

- Elemen utama

Kesalahan - Wilayah Tidak Valid

Kode masalah: INVALID_REGION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 supported Region.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 supported Region."
```

Menyelesaikan kesalahan

Perbarui nilai pasangan kunci-nilai kondisi untuk menyertakan Wilayah yang didukung. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- Nama dan kode wilayah

Kesalahan - Layanan tidak valid

Kode masalah: INVALID_SERVICE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Menyelesaikan kesalahan

Awalan layanan dalam kunci tindakan atau kondisi harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan cari awalannya di kalimat pertama.

Istilah terkait

- Layanan yang dikenal dan tindakan, sumber daya, dan kunci kondisinya

Kesalahan — Kunci kondisi layanan tidak valid

Kode masalah: INVALID_SERVICE_CONDITION_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Menyelesaikan kesalahan

Perbarui kunci dalam pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi yang diketahui untuk layanan. Nama kunci kondisi global dimulai dengan `aws` awalan. AWS layanan dapat menyediakan kunci khusus layanan yang menyertakan awalan layanan mereka. Untuk melihat awalan layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan.

Istilah terkait

- Kunci kondisi global
- Layanan yang dikenal dan tindakan, sumber daya, dan kunci kondisinya

Kesalahan - Layanan tidak valid dalam tindakan

Kode masalah: INVALID_SERVICE_IN_ACTION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Menyelesaikan kesalahan

Awalan layanan dalam tindakan harus sesuai dengan spesifikasi (termasuk kapitalisasi) untuk awalan layanan. Untuk melihat awalan layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan dan cari awalannya di kalimat pertama.

Istilah terkait

- Elemen tindakan
- Layanan yang dikenal dan tindakan mereka

Kesalahan — Variabel tidak valid untuk operator

Kode masalah: INVALID_VARIABLE_FOR_OPERATOR

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Menyelesaikan kesalahan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`. Kondisi mendukung variabel saat Anda menggunakan operator string atau operator ARN. Operator string termasuk`StringEquals`,`StringLike`, dan`StringNotLike`. Operator ARN termasuk `ArnEquals` dan`ArnLike`. Anda tidak dapat menggunakan variabel kebijakan dengan operator lain, seperti operator Numerik, Tanggal, Boolean, Biner, Alamat IP, atau Null.

Istilah terkait

- Menggunakan variabel kebijakan dalam elemen Kondisi
- Elemen kondisi

Kesalahan - Versi tidak valid

Kode masalah: INVALID_VERSION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid version: The version {{version}} is not valid. Use one of the following
 versions: {{versions}}
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The version {{version}} is not valid. Use one of the following
 versions: {{versions}}"
```

Menyelesaikan kesalahan

Elemen `Version` kebijakan menentukan aturan sintaks bahasa yang AWS digunakan untuk memproses kebijakan. Untuk menggunakan semua fitur kebijakan yang tersedia, sertakan `Version` elemen terbaru sebelum `Statement` elemen tersebut di semua kebijakan Anda.

```
"Version": "2012-10-17"
```

Istilah terkait

- Elemen versi

Kesalahan — Kesalahan sintaks Json

Kode masalah: JSON_SYNTAX_ERROR

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Menyelesaikan kesalahan

Kebijakan Anda menyertakan kesalahan sintaks. Periksa sintaks JSON Anda.

Istilah terkait

- Validator JSON
- Referensi elemen kebijakan JSON IAM
- Ikhtisar kebijakan JSON

Kesalahan — Kesalahan sintaks Json

Kode masalah: JSON_SYNTAX_ERROR

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Json syntax error: Fix the JSON syntax error.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Fix the JSON syntax error."
```

Menyelesaikan kesalahan

Kebijakan Anda menyertakan kesalahan sintaks. Periksa sintaks JSON Anda.

Istilah terkait

- Validator JSON
- Referensi elemen kebijakan JSON IAM
- Ikhtisar kebijakan JSON

Kesalahan - Tindakan hilang

Kode masalah: MISSING_ACTION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Menyelesaikan kesalahan

AWS Kebijakan JSON harus menyertakan `NotAction` elemen `Action` atau.

Istilah terkait

- Elemen tindakan
- NotAction elemen
- Ikhtisar kebijakan JSON

Kesalahan — Bidang ARN hilang

Kode masalah: MISSING_ARN_FIELD

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Menyelesaikan kesalahan

Semua bidang dalam sumber daya ARN harus sesuai dengan spesifikasi untuk jenis sumber daya yang diketahui. Untuk melihat format ARN yang diharapkan untuk layanan, lihat Tind akan, sumber daya, dan kunci kondisi untuk AWS layanan. Pilih nama layanan untuk melihat jenis sumber daya dan format ARN.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Wilayah ARN Hilang

Kode masalah: MISSING_ARN_REGION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Menyelesaikan kesalahan

ARN sumber daya untuk sebagian besar AWS layanan mengharuskan Anda menentukan Wilayah. Untuk tabel AWS layanan yang didukung di setiap Wilayah, lihat tabel Wilayah.

Istilah terkait

- Sumber daya kebijakan
- ARN Sumber Daya

- Nama dan kode wilayah

Kesalahan - Efek hilang

Kode masalah: MISSING_EFFECT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Menyelesaikan kesalahan

AWS Kebijakan JSON harus menyer `Effect` takan elemen dengan nilai **`Allow`** dan**`Deny`**.

Istilah terkait

- Elemen efek
- Ikhtisar kebijakan JSON

Kesalahan - Prinsip tidak ada

Kode masalah: MISSING_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing principal: Add a Principal element to the policy statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Menyelesaikan kesalahan

Resource-based Kebijakan harus mencakup suatu `Principal` elemen.

Misalnya, untuk menentukan akses untuk semua orang di AWS akun, gunakan prinsip berikut dalam kebijakan Anda:

```
"Principal": { "AWS": "123456789012" }
```

Istilah terkait

- Elemen utama
- Identity-based kebijakan dan kebijakan berbasis sumber daya

Kesalahan - Kualifikasi hilang

Kode masalah: MISSING_QUALIFIER

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing qualifier: The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Menyelesaikan kesalahan

Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi seperti sama atau kurang dari untuk membandingkan kondisi dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk permintaan yang menyertakan beberapa nilai untuk kunci kondisi tunggal, Anda harus menyertakan kondisi dalam tanda kurung seperti array (“Key2": [" Value2A”, “Value2B"]). Anda juga harus menggunakan operator `ForAllValues` atau `ForAnyValue` set dengan operator `StringLike` kondisi. Pengkualifikasi ini menambahkan fungsi operasi kumpulan ke operator kondisi sehingga Anda dapat menguji beberapa nilai permintaan terhadap beberapa nilai kondisi.

Istilah terkait

- Kunci konteks multinilai
- Elemen kondisi

AWS kebijakan yang dikelola dengan kesalahan ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan terkel AWS ola berikut menyertakan kualifikasi yang hilang untuk kunci kondisi dalam pernyataan kebijakan mereka. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, AWS Anda disarankan untuk menambahkan kualifikasi kunci `ForAllValues` atau `ForAnyValue` kondisi ke `Condition` elemen Anda.

- AWSGlueConsoleSageMakerNotebookFullAccess

Kesalahan - Sumber daya hilang

Kode masalah: MISSING_RESOURCE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Menyelesaikan kesalahan

Semua kebijakan kecuali kebijakan kepercayaan peran harus menyertakan `NotResource` elemen `Resource` atau.

Istilah terkait

- Elemen sumber daya
- NotResource elemen
- Identity-based kebijakan dan kebijakan berbasis sumber daya
- Ikhtisar kebijakan JSON

Kesalahan - Pernyataan hilang

Kode masalah: MISSING_STATEMENT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing statement: Add a statement to the policy
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a statement to the policy"
```

Menyelesaikan kesalahan

Kebijakan JSON harus menyertakan pernyataan.

Istilah terkait

- Elemen kebijakan JSON

Kesalahan — Null dengan jika ada

Kode masalah: NULL_WITH_IF_EXISTS

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Menyelesaikan kesalahan

Anda dapat menambahkan `IfExists` ke akhir nama operator kondisi apa pun kecuali operator `Null` kondisi. Gunakan operator ketentuan `Null` untuk memeriksa apakah kunci kondisi ada pada saat otorisasi. Gunakan `...ifExists` untuk mengatakan “Jika kunci kebijakan ada dalam konteks permintaan, proses kunci seperti yang ditentukan dalam kebijakan. Jika kuncinya tidak ada, evaluasi elemen ketentuan sebagai benar."

Istilah terkait

- ... IfExists operator kondisi
- Operator kondisi nol
- Elemen kondisi

Kesalahan — wildcard tindakan kesalahan sintaks SCP

Kode masalah: SCP_SYNTAX_ERROR_ACTION_WILDCARD

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) mendukung menentukan nilai dalam `NotAction` elemen `Action` atau. Namun, nilai-nilai ini dapat menyertakan wildcard (*) hanya di akhir string. Ini berarti bahwa Anda dapat menentukan `iam:Get*` tetapi tidak`iam:*role`.

Untuk menentukan beberapa tindakan, sarankan AWS Anda mencantumkannya satu per satu.

Istilah terkait

- Aksi dan NotAction elemen SCP
- Evaluasi SCP
- AWS Organizations kebijakan kontrol layanan
- Elemen kebijakan JSON IAM: Tindakan

Kesalahan — Prinsip kesalahan sintaks SCP

Kode masalah: SCP_SYNTAX_ERROR_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) tidak mendukung `NotPrincipal` elemen `Principal` atau.

Anda dapat menentukan Nama Sumber Daya Amazon (ARN) menggunakan kunci kondisi `aws:PrincipalArn` global di `Condition` elemen.

Istilah terkait

- Sintaks SCP
- Kunci kondisi global untuk kepala sekolah

Kesalahan - Sids Unik diperlukan

Kode masalah: UNIQUE_SIDS_REQUIRED

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Menyelesaikan kesalahan

Untuk beberapa jenis kebijakan, ID pernyataan harus unik. Elemen `Sid` (statement ID) memungkinkan Anda memasukkan pengidentifikasi opsional yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan nilai ID pernyataan untuk setiap pernyataan dalam array pernyataan menggunakan `SID` elemen. Dalam layanan yang memungkinkan Anda menentukan elemen ID, seperti SQS dan SNS, nil `Sid` ainya hanyalah sub-ID dari ID dokumen kebijakan. Misalnya, di IAM, nil `Sid` ainya harus unik dalam kebijakan JSON.

Istilah terkait

- Elemen kebijakan JSON IAM: Sid

Kesalahan — Tindakan yang tidak didukung dalam kebijakan

Kode masalah: UNSUPPORTED_ACTION_IN_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa tindakan tidak didukung dalam `Action` elemen dalam kebijakan berbasis sumber daya yang dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS Key Management Service tindakan tidak didukung dalam kebijakan bucket Amazon S3. Tentukan tindakan yang didukung oleh jenis sumber daya yang dilampirkan ke kebijakan berbasis sumber daya Anda.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan - Kombinasi elemen tidak didukung

Kode masalah: UNSUPPORTED_ELEMENT_COMBINATION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported element combination: The policy elements {{element1}} and {{element2}} can
 not be used in the same statement. Remove one of these elements.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy elements {{element1}} and {{element2}} can not be used in
 the same statement. Remove one of these elements."
```

Menyelesaikan kesalahan

Beberapa kombinasi elemen kebijakan JSON tidak dapat digunakan bersama. Misalnya, Anda tidak dapat menggunakan `Action` dan `NotAction` dalam pernyataan kebijakan yang sama. Pasangan lain yang saling eksklusif termasuk `Principal/NotPrincipal` dan`Resource/NotResource`.

Istilah terkait

- Referensi elemen kebijakan JSON IAM
- Ikhtisar kebijakan JSON

Kesalahan — Kunci kondisi global tidak didukung

Kode masalah: UNSUPPORTED_GLOBAL_CONDITION_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Menyelesaikan kesalahan

AWS tidak mendukung penggunaan kunci kondisi global yang ditentukan. Tergantung pada kasus penggunaan Anda, Anda dapat menggunakan kunci kondisi `aws:PrincipalArn` atau `aws:SourceArn` global. Misalnya, alih-alih`aws:ARN`, gunakan `aws:PrincipalArn` untuk membandingkan Nama Sumber Daya Amazon (ARN) dari prinsipal yang membuat permintaan dengan ARN yang Anda tentukan dalam kebijakan. Atau, gunakan kunci kondisi `aws:SourceArn` global untuk membandingkan Nama Sumber Daya Amazon (ARN) dari sumber daya yang membuat permintaan layanan-ke-layanan dengan ARN yang Anda tentukan dalam kebijakan.

Istilah terkait

- AWS kunci konteks kondisi global

Kesalahan - Prinsip tidak didukung

Kode masalah: UNSUPPORTED_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported principal: The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element."
```

Menyelesaikan kesalahan

`Principal`Elemen menentukan prinsip yang diizinkan atau ditolak akses ke sumber daya. Anda tidak dapat menggunakan elemen `Principal` dalam kebijakan berbasis-identitas IAM. Anda dapat menggunakannya dalam kebijakan kepercayaan untuk peran IAM dan kebijakan berbasis sumber daya. Resource-based kebijakan adalah kebijakan yang Anda sematkan langsung di sumber daya. Misalnya, Anda dapat menyematkan kebijakan di bucket Amazon S3 atau kunci AWS KMS.

Istilah terkait

- AWS Elemen kebijakan JSON: Principal
- Akses sumber daya lintas akun di IAM

Kesalahan — ARN sumber daya tidak didukung dalam kebijakan

Kode masalah: UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa ARN sumber daya tidak didukung dalam `Resource` elemen kebijakan berbasis sumber daya saat kebijakan dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS KMS ARN tidak

didukung dalam `Resource` elemen untuk kebijakan bucket Amazon S3. Tentukan ARN sumber daya yang didukung oleh jenis sumber daya yang dilampirkan ke kebijakan berbasis sumber daya Anda.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan - Sid tidak didukung

Kode masalah: UNSUPPORTED_SID

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Menyelesaikan kesalahan

Elemen `Sid` mendukung huruf besar, huruf kecil, dan angka.

Istilah terkait

- Elemen kebijakan JSON IAM: Sid

Kesalahan — Karakter pengganti tidak didukung di prinsip

Kode masalah: UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Menyelesaikan kesalahan

Struktur `Principal` elemen mendukung menggunakan pasangan kunci-nilai. Nilai utama yang ditentukan dalam kebijakan menyertakan wildcard (*). Anda tidak dapat menyertakan wildcard dengan kunci utama yang Anda tentukan. Misalnya, ketika Anda menentukan pengguna dalam `Principal` elemen, Anda tidak dapat menggunakan wildcard untuk berarti “semua pengguna”. Anda harus memberi nama pengguna atau pengguna tertentu. Demikian pula, ketika Anda menentukan sesi peran yang diasumsikan, Anda tidak dapat menggunakan wildcard untuk berarti “semua sesi”. Anda harus memberi nama sesi tertentu. Anda juga tidak dapat menggunakan wildcard untuk mencocokkan bagian dari nama atau ARN.

Untuk mengatasi temuan ini, hapus wildcard dan berikan prinsip yang lebih spesifik.

Istilah terkait

- AWS Elemen kebijakan JSON: Principal

Kesalahan - Tidak ada penjepit dalam variabel

Kode masalah: MISSING_BRACE_IN_VARIABLE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurung kurung keriting ({ }). Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan.

Untuk mengatasi temuan ini, tambahkan penjepit yang hilang untuk memastikan ada set kawat gigi pembuka dan penutup penuh.

Istilah terkait

- Elemen kebijakan IAM: Variabel

Kesalahan - Kutipan hilang dalam variabel

Kode masalah: MISSING_QUOTE_IN_VARIABLE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Menyelesaikan kesalahan

Ketika Anda menambahkan variabel ke kebijakan Anda, Anda dapat menentukan nilai default untuk variabel. Jika variabel tidak ada, AWS gunakan teks default yang Anda berikan.

Untuk menambahkan nilai default ke variabel, sertai nilai default dengan tanda kutip tunggal (`' '`), dan pisahkan teks variabel serta nilai default dengan koma dan spasi (`, `).

Misalnya, jika kepala sekolah diberi tag`team=yellow`, mereka dapat mengakses bucket `amzn-s3-demo-bucket` Amazon S3 dengan nama tersebut`amzn-s3-demo-bucket-yellow`. Kebijakan dengan sumber daya ini mungkin memungkinkan anggota tim mengakses sumber daya mereka sendiri, tetapi tidak untuk tim lain. Untuk pengguna tanpa tag tim, Anda dapat menetapkan nilai default`company-wide`. Pengguna ini hanya dapat mengakses `amzn-s3-demo-bucket-company-wide` bucket tempat mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Istilah terkait

- Elemen kebijakan IAM: Variabel

Kesalahan - Ruang tidak didukung dalam variabel

Kode masalah: UNSUPPORTED_SPACE_IN_VARIABLE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurung kurung keriting ({ }). Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan. Meskipun Anda dapat menyertakan spasi ketika Anda menentukan variabel default, Anda tidak dapat menyertakan spasi dalam nama variabel.

Istilah terkait

- Elemen kebijakan IAM: Variabel

Kesalahan - Variabel kosong

Kode masalah: EMPTY_VARIABLE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Menyelesaikan kesalahan

Struktur variabel kebijakan mendukung penggunaan $ awalan diikuti oleh sepasang kurung kurung keriting ({ }). Di dalam ${ } karakter, sertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan.

Istilah terkait

- Elemen kebijakan IAM: Variabel

Kesalahan — Variabel tidak didukung dalam elemen

Kode masalah: VARIABLE_UNSUPPORTED_IN_ELEMENT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Menyelesaikan kesalahan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`.

Istilah terkait

- Elemen kebijakan IAM: Variabel

Kesalahan - Variabel tidak didukung dalam versi

Kode masalah: VARIABLE_UNSUPPORTED_IN_VERSION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Menyelesaikan kesalahan

Untuk menggunakan variabel kebijakan, Anda harus menyertakan `Version` elemen dan mengaturnya ke versi yang mendukung variabel kebijakan. Variabel diperkenalkan dalam versi `2012-10-17`. Versi sebelumnya dari bahasa kebijakan tidak mendukung variabel kebijakan. Jika Anda tidak menyetel `Version` ke `2012-10-17` atau yang lebih baru, variabel seperti `${aws:username}` diperlakukan sebagai string literal dalam kebijakan.

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, dibuat saat Anda mengubah kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola.

Istilah terkait

- Elemen kebijakan IAM: Variabel
- Elemen kebijakan JSON IAM: Versi

Kesalahan - Alamat IP pribadi

Kode masalah: PRIVATE_IP_ADDRESS

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Menyelesaikan kesalahan

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik. Anda menerima kesalahan ini jika kebijakan Anda hanya mengizinkan alamat IP pribadi. Dalam hal ini, kondisinya tidak akan pernah cocok.

- aws: kunci kondisi SourceIp global
- Elemen kebijakan JSON IAM: Kondisi

Kesalahan - Pribadi NotIpAddress

Kode masalah: PRIVATE_NOT_IP_ADDRESS

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Menyelesaikan kesalahan

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik. Anda menerima kesalahan ini ketika Anda menggunakan operator `NotIpAddress` kondisi dan hanya mencantumkan alamat IP pribadi. Dalam hal ini, kondisinya akan selalu cocok dan tidak efektif.

- aws: kunci kondisi SourceIp global
- Elemen kebijakan JSON IAM: Kondisi

Kesalahan - Ukuran kebijakan melebihi kuota SCP

Kode masalah: POLICY_SIZE_EXCEEDS_SCP_QUOTA

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol layanan (SCP) mendukung menentukan nilai dalam `NotAction` elemen `Action` atau. Namun, nilai-nilai ini dapat menyertakan wildcard (*) hanya di akhir string. Ini berarti bahwa Anda dapat menentukan `iam:Get*` tetapi tidak`iam:*role`.

Untuk menentukan beberapa tindakan, sarankan AWS Anda mencantumkannya satu per satu.

Istilah terkait

- Kuota untuk AWS Organisasi
- AWS Organizations kebijakan kontrol layanan

Kesalahan — Format utama layanan tidak valid

Kode masalah: INVALID_SERVICE_PRINCIPAL_FORMAT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Menyelesaikan kesalahan

Nilai dalam pasangan kunci-nilai kondisi harus sesuai dengan format prinsip layanan yang ditentukan.

Pengguna utama layanan adalah pengidentifikasi yang digunakan untuk memberikan izin layanan. Anda dapat menentukan prinsip layanan dalam `Principal` elemen atau sebagai nilai untuk beberapa kunci kondisi global dan kunci khusus layanan. Prinsip layanan ditentukan oleh setiap layanan.

Pengidentifikasi untuk prinsipal layanan mencakup nama layanan, dan biasanya dalam format berikut dalam semua huruf kecil:

`service-name``.amazonaws.com`

Beberapa kunci khusus layanan mungkin menggunakan format yang berbeda untuk prinsipal layanan. Misalnya, kunci `kms:ViaService` kondisi memerlukan format berikut untuk prinsipal layanan dalam semua huruf kecil:

`service-name.AWS_region``.amazonaws.com`

Istilah terkait

- Prinsip layanan
- AWS kunci kondisi global
- `kms:ViaService`kunci kondisi

Kesalahan - Kunci tag hilang dalam kondisi

Kode masalah: MISSING_TAG_KEY_IN_CONDITION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Menyelesaikan kesalahan

Untuk mengontrol akses berdasarkan tanda, Anda memberikan informasi tanda di elemen ketentuan dari kebijakan.

Misalnya, untuk mengontrol akses ke AWS sumber daya, Anda menyertakan kunci `aws:ResourceTag` kondisi. Kunci ini membutuhkan format`aws:ResourceTag/`*`tag-key`*. Untuk menentukan kunci tag `owner` dan nilai tag `JaneDoe` dalam suatu kondisi, gunakan format berikut.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Istilah terkait

- Mengontrol akses menggunakan tag
- Kondisi
- Kunci kondisi global
- AWS kunci kondisi layanan

Kesalahan — Format vpc tidak valid

Kode masalah: INVALID_VPC_FORMAT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Menyelesaikan kesalahan

K `aws:SourceVpc` unci kondisi harus menggunakan awalan `vpc-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, `vpc-11223344556677889` atau`vpc-12345678`.

Istilah terkait

- AWS kunci kondisi global: aws: SourceVpc

Kesalahan — Format vpce tidak valid

Kode masalah: INVALID_VPCE_FORMAT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Menyelesaikan kesalahan

K `aws:SourceVpce` unci kondisi harus menggunakan awalan `vpce-` diikuti oleh 8 atau 17 karakter alfanumerik, misalnya, `vpce-11223344556677889` atau`vpce-12345678`.

Istilah terkait

- AWS kunci kondisi global: aws: SourceVpce

Kesalahan - Prinsip federasi tidak didukung

Kode masalah: FEDERATED_PRINCIPAL_NOT_SUPPORTED

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Menyelesaikan kesalahan

`Principal`Elemen menggunakan prinsipal federasi untuk kebijakan kepercayaan yang dilampirkan ke peran IAM untuk menyediakan akses melalui federasi identitas. Kebijakan identitas dan kebijakan berbasis sumber daya lainnya tidak mendukung penyedia identitas federasi dalam elemen. `Principal` Misalnya, Anda tidak dapat menggunakan prinsip SAML dalam kebijakan bucket Amazon S3. Ubah `Principal` elemen ke tipe utama yang didukung.

Istilah terkait

- Menciptakan peran untuk federasi identitas
- Elemen kebijakan JSON: Principal

Kesalahan — Tindakan tidak didukung untuk kunci kondisi

Kode masalah: UNSUPPORTED_ACTION_FOR_CONDITION_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Menyelesaikan kesalahan

Pastikan bahwa kunci kondisi dalam `Condition` elemen pernyataan kebijakan berlaku untuk setiap tindakan dalam `Action` elemen. Untuk memastikan bahwa tindakan yang Anda tentukan secara efektif diizinkan atau ditolak oleh kebijakan Anda, Anda harus memindahkan tindakan yang tidak didukung ke pernyataan lain tanpa kunci kondisi.

> **Note**
>
> Jika `Action` elemen memiliki tindakan dengan wildcard, IAM Access Analyzer tidak mengevaluasi tindakan tersebut untuk kesalahan ini.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan — Tindakan yang tidak didukung dalam kebijakan

Kode masalah: UNSUPPORTED_ACTION_IN_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa tindakan tidak didukung dalam `Action` elemen dalam kebijakan berbasis sumber daya yang dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS Key Management Service tindakan tidak didukung dalam kebijakan bucket Amazon S3. Tentukan tindakan yang didukung oleh jenis sumber daya yang dilampirkan ke kebijakan berbasis sumber daya Anda.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan — ARN sumber daya tidak didukung dalam kebijakan

Kode masalah: UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Menyelesaikan kesalahan

Beberapa ARN sumber daya tidak didukung dalam `Resource` elemen kebijakan berbasis sumber daya saat kebijakan dilampirkan ke jenis sumber daya yang berbeda. Misalnya, AWS KMS ARN tidak didukung dalam `Resource` elemen untuk kebijakan bucket Amazon S3. Tentukan ARN sumber daya yang didukung oleh jenis sumber daya yang dilampirkan ke kebijakan berbasis sumber daya Anda.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan — Kunci kondisi tidak didukung untuk prinsipal layanan

Kode masalah: UNSUPPORTED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Menyelesaikan kesalahan

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengidentifikasi untuk layanan. Anda tidak dapat menggunakan beberapa kunci kondisi dengan prinsipal layanan tertentu. Misalnya, Anda tidak dapat menggunakan kunci `aws:PrincipalOrgID` kondisi dengan prinsipal layanan`cloudfront.amazonaws.com`. Anda harus menghapus kunci kondisi yang tidak berlaku untuk prinsipal layanan di `Principal` elemen.

Istilah terkait

- Prinsip layanan
- Elemen kebijakan JSON: Principal

Kesalahan — Kesalahan sintaks kebijakan kepercayaan peran notprincipal

Kode masalah: ROLE_TRUST_POLICY_SYNTAX_ERROR_NOTPRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada peran IAM. Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan

pengguna gabungan) yang dapat memegang peran tersebut. Kebijakan kepercayaan peran tidak mendukung`NotPrincipal`. Perbarui kebijakan untuk menggunakan `Principal` elemen sebagai gantinya.

Istilah terkait

- Elemen kebijakan JSON: Principal
- Elemen kebijakan JSON: NotPrincipal

Kesalahan — Kebijakan kepercayaan peran tidak didukung wildcard di prinsip

Kode masalah: ROLE_TRUST_POLICY_UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada peran IAM. Kebijakan kepercayaan menentukan entitas utama mana (akun, pengguna, peran, dan pengguna federasi) yang dapat mengambil peran tersebut. `"Principal:" "*"`tidak didukung dalam `Principal` elemen kebijakan kepercayaan peran. Ganti wildcard dengan nilai pokok yang valid.

Istilah terkait

- Elemen kebijakan JSON: Principal

Kesalahan — Sumber daya kesalahan sintaks kebijakan kepercayaan peran

Kode masalah: ROLE_TRUST_POLICY_SYNTAX_ERROR_RESOURCE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Menyelesaikan kesalahan

Kebijakan kepercayaan peran adalah kebijakan berbasis sumber daya yang melekat pada peran IAM. Kebijakan kepercayaan menentukan entitas prinsipal mana (akun, pengguna, peran, dan pengguna gabungan) yang dapat memegang peran tersebut. Kebijakan kepercayaan peran berlaku untuk peran yang dilampirkan. Anda tidak dapat menentukan `NotResource` elemen `Resource` atau dalam kebijakan kepercayaan peran. Hapus `NotResource` elemen `Resource` atau.

- Elemen kebijakan JSON: Sumber daya
- Elemen kebijakan JSON: NotResource

Kesalahan - Jenis rentang IP yang tidak cocok

Kode masalah: TYPE_MISMATCH_IP_RANGE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Menyelesaikan kesalahan

Perbarui teks untuk menggunakan tipe data operator kondisi alamat IP, dalam format CIDR.

Istilah terkait

- Operator kondisi alamat IP
- Elemen kebijakan JSON IAM: Operator kondisi

Kesalahan - Tindakan hilang untuk kunci kondisi

Kode masalah: MISSING_ACTION_FOR_CONDITION_KEY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Menyelesaikan kesalahan

Kunci kondisi dalam `Condition` elemen pernyataan kebijakan tidak dievaluasi kecuali tindakan yang ditentukan ada dalam `Action` elemen. Untuk memastikan bahwa kunci kondisi yang Anda tentukan secara efektif diizinkan atau ditolak oleh kebijakan Anda, tambahkan tindakan ke `Action` elemen.

Istilah terkait

- Elemen kebijakan JSON: Tindakan

Kesalahan — Sintaks utama federasi tidak valid dalam kebijakan kepercayaan peran

Kode masalah: INVALID_FEDERATED_PRINCIPAL_SYNTAX_IN_ROLE_TRUST_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Menyelesaikan kesalahan

Nilai utama menentukan prinsipal federasi yang tidak sesuai dengan format yang diharapkan. Perbarui format prinsipal federasi ke nama domain yang valid atau ARN metadata SAML.

Istilah terkait

- Pengguna dan peran federasi

Kesalahan — Tindakan yang tidak cocok untuk prinsipal

Kode masalah: MISMATCHED_ACTION_FOR_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Menyelesaikan kesalahan

Tindakan yang ditentukan dalam `Action` elemen pernyataan kebijakan tidak valid dengan prinsip yang ditentukan dalam `Principal` elemen. Misalnya, Anda tidak dapat menggunakan prinsipal penyedia SAML dengan `sts:AssumeRoleWithWebIdentity` tindakan tersebut. Anda harus menggunakan prinsip penyedia SAML dengan `sts:AssumeRoleWithSAML` tindakan atau menggunakan prinsip penyedia OIDC dengan tindakan tersebut`sts:AssumeRoleWithWebIdentity`.

Istilah terkait

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Kesalahan — Tindakan hilang untuk kebijakan kepercayaan peran di mana saja

Kode masalah: MISSING_ACTION_FOR_ROLES_ANYWHERE_TRUST_POLICY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Menyelesaikan kesalahan

Agar IAM Roles Anywhere dapat mengambil peran dan memberikan kredentif sementara, peran AWS tersebut harus mempercayai prinsip layanan IAM Roles Anywhere. Prinsip layanan IAM Roles Anywhere memerlukan`sts:AssumeRole`,`sts:SetSourceIdentity`, dan `sts:TagSession` izin untuk mengambil peran. Jika ada izin yang hilang, Anda harus menambahkannya ke kebijakan Anda.

Istilah terkait

- Model kepercayaan dalam Peran di M AWS Identity and Access Management ana Saja

Kesalahan — Ukuran kebijakan melebihi kuota RCP

Kode masalah: POLICY_SIZE_EXCEEDS_RCP_QUOTA

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds RCP quota: The {{policySize}} characters in the resource control
 policy (RCP) exceed the {{policySizeQuota}} character maximum for RCPs. We recommend
 that you use multiple granular policies.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the resource control policy (RCP)
 exceed the {{policySizeQuota}} character maximum for RCPs. We recommend that you use
 multiple granular policies."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol sumber daya (RCP) mendukung menentukan nilai dalam `Action` elemen. Namun, nilai-nilai ini dapat menyertakan wildcard (*) hanya di akhir string. Ini berarti bahwa Anda dapat menentukan `s3:Get*` tetapi tidak`s3:*Object`.

Untuk menentukan beberapa tindakan, sarankan AWS Anda mencantumkannya satu per satu.

Istilah terkait

- Kuota untuk AWS Organizations
- AWS Organizations kebijakan kontrol sumber daya

Kesalahan — Prinsip kesalahan sintaks RCP

Kode masalah: RCP_SYNTAX_ERROR_PRINCIPAL

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP syntax error principal: The Principal element contents are not valid. RCPs only
 support specifying all principals ("*") in the Principal element. The NotPrincipal
 element is not supported for RCPs.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element contents are not valid. RCPs only support
 specifying all principals ("*") in the Principal element. The NotPrincipal element is
 not supported for RCPs."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol sumber daya (RCP) hanya mendukung penentuan semua prinsipal (" * ") dalam elemen. `Principal` `NotPrincipal`Elemen tidak didukung untuk RCP.

Istilah terkait

- Sintaks RCP
- Properti prinsipal

Kesalahan — Kesalahan sintaks RCP memungkinkan

Kode masalah: RCP_SYNTAX_ERROR_ALLOW

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP syntax error allow: RCPs only support specifying all principals ("*") in the
 Principal element, all resources ("*") in the Resource element, and no Condition
 element with an effect of Allow.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "RCPs only support specifying all principals ("*") in the Principal
 element, all resources ("*") in the Resource element, and no Condition element with an
 effect of Allow."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol sumber daya (RCP) hanya mendukung penentuan semua prinsipal (" * ") dalam `Principal` elemen dan semua sumber daya (" * ") dalam elemen. `Resource` `Condition`Elemen dengan efek tidak `Allow` didukung untuk RCP.

Istilah terkait

- Sintaks RCP

- Properti prinsipal
- Properti sumber daya

Kesalahan — Kesalahan sintaks RCP NotAction

Kode masalah: RCP_SYNTAX_ERROR_NOTACTION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP syntax error NotAction: RCPs do not support the NotAction element. Update to use
 the Action element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "RCPs do not support the NotAction element. Update to use the Action
 element."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol sumber daya (RCP) tidak mendukung `NotAction` elemen. Gunakan `Action` elemen.

Istilah terkait

- Sintaks RCP
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: NotAction

Kesalahan — tindakan kesalahan sintaks RCP

Kode masalah: RCP_SYNTAX_ERROR_ACTION

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP syntax error action: RCPs only support specifying select service prefixes in the
 Action element. Learn more here.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "RCPs only support specifying select service prefixes in the Action
 element. Learn more here."
```

Menyelesaikan kesalahan

AWS Organizations kebijakan kontrol sumber daya (RCP) hanya mendukung penentuan awalan layanan pilih dalam elemen. `Action`

Istilah terkait

- Sintaks RCP
- Daftar R Layanan AWS CP yang mendukung

Kesalahan — Akun ARN hilang

Kode masalah: MISSING_ARN_ACCOUNT

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing ARN account: The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id."
```

Menyelesaikan kesalahan

Sertakan ID akun di sumber daya ARN. ID akun adalah bilangan bulat 12 digit. Untuk mempelajari cara melihat ID akun Anda, lihat M enemukan ID AWS akun Anda.

Istilah terkait

- Sumber daya kebijakan
- Pengidentifikasi Akun
- ARN Sumber Daya
- AWS sumber daya layanan dengan format ARN

Kesalahan — Nilai kunci kms tidak valid

Kode masalah: INVALID_KMS_KEY_VALUE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid kms key value: The {{key}} condition key value must be a valid KMS key ARN.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{key}} condition key value must be a valid KMS key ARN."
```

Menyelesaikan kesalahan

AWS KMS key ARN (Nama Sumber Daya Amazon) adalah pengenal unik yang memenuhi syarat untuk kunci KMS. Kunci ARN mencakup Akun AWS, Wilayah, dan ID kunci. ARN kunci mengikuti format ini:

`arn:aws:kms:region:account-id:key/key-id`

Istilah terkait

- Temukan ID kunci dan kunci ARN
- Kunci ARN
- AWS kunci konteks kondisi global

Kesalahan - Penggunaan variabel terlalu permisif

Kode masalah: VARIABLE_USAGE_TOO_PERMISSIVE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Variable usage too permissive: Overly permissive use of policy variable for the {{key}}
 condition key. Use the policy variable preceded by 6 consecutive characters.
```

```
The policy variable is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and policy variables can be evaluated as effective wildcards. Therefore
 policy variables are not allowed to be used with sensitive keys. Refer to https://
docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html for the
 list of sensitive keys.
```

```
Overly permissive use of policy variable with aws:userID. Use the policy variable on
 the right side of colon or as the only character on the left side of a colon.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Overly permissive use of policy variable for the {{key}} condition
 key. Use the policy variable preceded by 6 consecutive characters."
```

```
"findingDetails": "The policy variable is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and policy variables can be evaluated as
```

```
 effective wildcards. Therefore policy variables are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of policy variable with aws:userID. Use the
 policy variable on the right side of colon or as the only character on the left side
 of a colon."
```

Menyelesaikan kesalahan

Ada tiga kemungkinan pesan penemuan untuk kesalahan ini.

- Untuk pesan kesalahan pertama, ubah penggunaan variabel kebijakan agar lebih spesifik. Tambahkan setidaknya 6 karakter berturut-turut sebelum variabel kebijakan untuk mengurangi cakupan izin. Misalnya, alih-alih menggunakan`${aws:username}`, gunakan `prefix-${aws:username}` ataumyapp-`${aws:username}`. Ini memastikan bahwa variabel kebijakan tidak memberikan akses yang terlalu luas.
- Untuk pesan kesalahan kedua, hapus variabel kebijakan dari kunci kondisi yang ditentukan. Variabel kebijakan dapat bertindak sebagai wildcard yang efektif dan tidak diizinkan dengan kunci kondisi sensitif karena alasan keamanan. Sebagai gantinya, gunakan nilai statis tertentu atau pertimbangkan untuk merestrukturisasi kebijakan Anda untuk menggunakan kunci kondisi tidak sensitif yang mendukung variabel kebijakan.
- Untuk pesan kesalahan ketiga, ubah penggunaan variabel `aws:userID` kebijakan agar lebih ketat. Tempatkan variabel kebijakan di sisi kanan titik dua (setelah ID akun) atau gunakan sebagai satu-satunya karakter di sisi kiri titik dua. Misalnya, gunakan `AIDACKCEVSQ6C2EXAMPLE:${aws:userid}` atau `${aws:userid}:*` bukannya`${aws:userid}`.

Istilah terkait

- Elemen kebijakan IAM: Variabel dan tag
- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global

Kesalahan - Penggunaan wildcard terlalu permisif

Kode masalah: WILDCARD_USAGE_TOO_PERMISSIVE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Wildcard usage too permissive: Overly permissive use of wildcard for the {{key}}
 condition key. Use the wildcard preceded by 6 consecutive characters.
```

```
The wildcard is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and wildcards are not allowed to be used with sensitive keys. Refer
 to https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-
keys.html for the list of sensitive keys.
```

```
Overly permissive use of wildcard with aws:userID. Use the wildcard on the right side
 of colon or as the only character on the left side of a colon.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Overly permissive use of wildcard for the {{key}} condition key. Use
 the wildcard preceded by 6 consecutive characters."
```

```
"findingDetails": "The wildcard is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and wildcards are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of wildcard with aws:userID. Use the wildcard
 on the right side of colon or as the only character on the left side of a colon."
```

Menyelesaikan kesalahan

Ada tiga kemungkinan pesan penemuan untuk kesalahan ini.

- Untuk pesan kesalahan pertama, buat penggunaan wildcard Anda lebih spesifik dengan menambahkan setidaknya 6 karakter berturut-turut sebelum wildcard. Misalnya, alih-alih menggunakan`*`, gunakan `prefix-*` atau`prefix-*-suffix`. Ini mengurangi ruang lingkup kondisi dan mengikuti prinsip hak istimewa terkecil.

- Untuk pesan kesalahan kedua, hapus wildcard dari kunci kondisi yang ditentukan. Wildcard tidak diizinkan dengan kunci kondisi sensitif untuk alasan keamanan. Ganti wildcard dengan nilai tertentu yang sesuai dengan pola akses yang Anda inginkan, atau pertimbangkan untuk menggunakan kunci kondisi non-sensitif yang berbeda yang mendukung wildcard.
- Untuk pesan kesalahan ketiga, ubah penggunaan `aws:userID` wildcard Anda agar lebih ketat. Tempatkan wildcard di sisi kanan titik dua (setelah ID akun) atau gunakan sebagai satu-satunya karakter di sisi kiri titik dua. Misalnya, gunakan `AIDACKCEVSQ6C2EXAMPLE:*` atau `*:*` bukannya`*`.

Istilah terkait

- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global
- Pengidentifikasi IAM

Kesalahan — DynamoDB tidak valid mengembalikan kapasitas yang dikonsumsi

Kode masalah: DYNAMODB_INVALID_RETURN_CONSUMED_CAPACITY

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB invalid return consumed capacity: The value for
 dynamodb:ReturnConsumedCapacity is not valid. Update {{value}} to either TOTAL or
 NONE.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The value for dynamodb:ReturnConsumedCapacity is not valid. Update
 {{value}} to either TOTAL or NONE."
```

Menyelesaikan kesalahan

Perbarui nilai kondisi ke salah satu dari dua nilai yang diberikan DynamoDB: `TOTAL` atau. `NONE` Nilai peka huruf besar- `StringEquals` kecil di bawah dan`StringLike`, jadi varian huruf kecil seperti tidak `total` valid. `StringEqualsIgnoreCase`Operator menerima casing apa pun. Kondisi yang menentukan nilai yang tidak valid tidak pernah cocok dengan permintaan apa pun, sehingga pernyataan tidak berpengaruh.

Kesalahan — nilai pengembalian DynamoDB tidak valid

Kode masalah: DYNAMODB_INVALID_RETURN_VALUES

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB invalid return values: The value for dynamodb:ReturnValues is not valid.
 Update {{value}} to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The value for dynamodb:ReturnValues is not valid. Update {{value}}
 to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW."
```

Menyelesaikan kesalahan

Perbarui nilai kondisi ke salah satu dari lima nilai yang diberikan DynamoDB:`NONE`,,, `ALL_OLD` `UPDATED_OLD`ALL_NEW, atau. `UPDATED_NEW` Nilai peka huruf besar- `StringEquals` kecil di bawah dan`StringLike`, jadi varian huruf kecil seperti tidak `updated_new` valid. `StringEqualsIgnoreCase`Operator menerima casing apa pun. Kondisi yang menentukan nilai yang tidak valid tidak pernah cocok dengan permintaan apa pun, sehingga pernyataan tidak berpengaruh.

Kesalahan — nilai pilihan DynamoDB tidak valid

Kode masalah: DYNAMODB_INVALID_SELECT_VALUE

Jenis pencarian: ERR OR

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB invalid select value: The value for dynamodb:Select is not valid. Update
 {{value}} to one of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or
 COUNT.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The value for dynamodb:Select is not valid. Update {{value}} to one
 of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or COUNT."
```

Menyelesaikan kesalahan

Perbarui nilai kondisi ke salah satu dari empat nilai yang diberikan DynamoDB:`ALL_ATTRIBUTES`,, `ALL_PROJECTED_ATTRIBUTESSPECIFIC_ATTRIBUTES`, atau. `COUNT` Nilai peka huruf besar- `StringEquals` kecil di bawah dan`StringLike`, jadi varian huruf kecil seperti tidak `specific_attributes` valid. `StringEqualsIgnoreCase`Operator menerima casing apa pun. Kondisi yang menentukan nilai yang tidak valid tidak pernah cocok dengan permintaan apa pun, sehingga pernyataan tidak berpengaruh.

Peringatan Umum - Buat SLR dengan NotResource

Kode masalah: CREATE_SLR_WITH_NOT_RESOURCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` dalam kebijakan dengan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Umum - Buat SLR dengan bintang beraksi dan NotResource

Kode masalah: CREATE_SLR_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Kebijakan dengan wildcard (*) di `Action` dan yang menyertakan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Umum - Buat SLR dengan dan NotAction NotResource

Kode masalah: CREATE_SLR_WITH_NOT_ACTION_AND_NOT_RESOURCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `NotAction` elemen dengan `NotResource` elemen dapat memungkinkan pembuatan peran terkait layanan

yang tidak diinginkan untuk beberapa sumber daya. AWS menyarankan agar Anda menulis ulang kebijakan untuk mengiz `iam:CreateServiceLinkedRole` inkan daftar terbatas ARN di `Resource` elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: NotAction
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Umum - Buat SLR dengan bintang di sumber daya

Kode masalah: CREATE_SLR_WITH_STAR_IN_RESOURCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `iam:CreateServiceLinkedRole` kebijakan dengan wildcard (*) di `Resource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya.

AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: Sumber

AWS kebijakan yang dikelola dengan peringatan umum ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk pengguna kuat dalam akun Anda. Kebijakan terkel AWS ola berikut menyediakan akses pengguna yang kuat dan memberikan izin untuk membuat peran terkait layanan untuk layanan apa pun AWS . AWS menyarankan agar Anda melampirkan kebijakan ter AWS kelola berikut hanya ke identitas IAM yang Anda anggap pengguna utama.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Kebijakan AWS terkelola ini memberikan izin untuk digunakan oleh peran AWS Organizations terkait layanan. Peran ini memungkinkan Organisasi untuk membuat peran terkait layanan tambahan untuk layanan lain di organisasi Anda AWS .

Peringatan Umum - Buat SLR dengan bintang dalam aksi dan sumber daya

Kode masalah: CREATE_SLR_WITH_STAR_IN_ACTION_AND_RESOURCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Kebijakan dengan wildcard (*) di `Resource` elemen `Action` dan dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. Hal ini memungkinkan pembuatan peran terkait layanan saat Anda menentukan`"Action": "*"`,`"Action": "iam:*"`, atau. `"Action": "iam:Create*"` AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: Sumber

AWS kebijakan yang dikelola dengan peringatan umum ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan ter AWS kelola berikut menyediakan akses administrator dan memberikan izin untuk membuat peran terkait layanan untuk layanan apa pun AWS . AWS menyarankan agar Anda melampirkan kebijakan ter AWS kelola berikut hanya ke identitas IAM yang Anda anggap administrator.

- AdministratorAccess
- IAMFullAccess

Peringatan Umum - Buat SLR dengan bintang di sumber daya dan NotAction

Kode masalah: CREATE_SLR_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan umum

Tind `iam:CreateServiceLinkedRole` akan memberikan izin untuk membuat peran IAM yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Menggunakan `NotAction` elemen dalam kebijakan dengan wildcard (*) di `Resource` elemen dapat memungkinkan pembuatan peran terkait layanan yang tidak diinginkan untuk beberapa sumber daya. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Anda juga dapat `iam:CreateServiceLinkedRole` menambahkan `NotAction` elemen.

- CreateServiceLinkedRole operasi
- Elemen kebijakan JSON IAM: NotAction
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: Sumber

Peringatan Umum — Kunci kondisi global yang tidak digunakan lagi

Kode masalah: DEPRECATED_GLOBAL_CONDITION_KEY

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Menyelesaikan peringatan umum

Kebijakan ini menyertakan kunci kondisi global yang tidak digunakan lagi. Perbarui kunci kondisi dalam pasangan kunci-nilai kondisi untuk menggunakan kunci kondisi global yang didukung.

- Kunci kondisi global

Peringatan Umum - Nilai tanggal tidak valid

Kode masalah: INVALID_DATE_VALUE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Menyelesaikan peringatan umum

Waktu Unix Epoch menggambarkan titik waktu yang telah berlalu sejak 1 Januari 1970, dikurangi detik kabisat. Waktu zaman mungkin tidak ditentukan dengan waktu yang tepat yang Anda harapkan.

AWS merekomendasikan agar Anda menggunakan standar W3C untuk format tanggal dan waktu. Misalnya, Anda dapat menentukan tanggal lengkap, seperti YYYY-MM-DD (1997-07-16), atau Anda juga dapat menambahkan waktu ke tanggal kedua, seperti YYYY-MM-DDThh:mm:ssTZD (199 T19:20:30 7-07-16 + 01:00).

- Format Tanggal dan Waktu W3C
- Elemen kebijakan JSON IAM: Versi
- aws: kunci kondisi CurrentTime global

Peringatan Umum — Referensi peran tidak valid

Kode masalah: INVALID_ROLE_REFERENCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Menyelesaikan peringatan umum

AWS menyarankan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk peran IAM alih-alih ID utamanya. Ketika IAM menyimpan kebijakan, itu akan mengubah ARN menjadi ID utama untuk peran yang ada. AWS termasuk tindakan pencegahan keamanan. Jika seseorang menghapus dan membuat ulang peran tersebut, ia akan memiliki ID baru, dan kebijakan tidak akan cocok dengan ID peran baru tersebut.

- Menentukan prinsip: peran IAM
- ARN IAM

- ID unik IAM

Peringatan Umum — Referensi pengguna tidak valid

Kode masalah: INVALID_USER_REFERENCE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Menyelesaikan peringatan umum

AWS menyarankan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk pengguna IAM alih-alih ID utamanya. Ketika IAM menyimpan kebijakan, itu akan mengubah ARN menjadi ID utama untuk pengguna yang ada. AWS termasuk tindakan pencegahan keamanan. Jika seseorang menghapus dan membuat ulang pengguna, pengguna tersebut akan memiliki ID baru, dan kebijakan tersebut tidak akan cocok dengan ID pengguna baru.

- Menentukan prinsipal: pengguna IAM
- ARN IAM
- ID unik IAM

Peringatan Umum - Versi yang hilang

Kode masalah: MISSING_VERSION

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Menyelesaikan peringatan umum

AWS menyarankan agar Anda menyertakan `Version` parameter opsional dalam kebijakan Anda. Jika Anda tidak menyertakan elemen Versi, nilai default adalah`2012-10-17`, tetapi fitur yang lebih baru, seperti variabel kebijakan, tidak akan berfungsi dengan kebijakan Anda. Misalnya, variabel seperti `${aws:username}` tidak diakui sebagai variabel dan diperlakukan sebagai string literal di dalam kebijakan.

- Elemen kebijakan JSON IAM: Versi

Peringatan Umum — Sids Unik direkomendasikan

Kode masalah: UNIQUE_SIDS_RECOMMENDED

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Menyelesaikan peringatan umum

AWS merekomendasikan agar Anda menggunakan ID pernyataan unik. Elemen `Sid` (statement ID) memungkinkan Anda memasukkan pengidentifikasi opsional yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan nilai ID pernyataan untuk setiap pernyataan dalam array pernyataan menggunakan `SID` elemen.

Istilah terkait

- Elemen kebijakan JSON IAM: Sid

Peringatan Umum - Wildcard tanpa operator seperti

Kode masalah: WILDCARD_WITHOUT_LIKE_OPERATOR

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Menyelesaikan peringatan umum

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Ketika Anda menentukan nilai kondisi yang menggunakan wildcard (*,?) , Anda harus menggunakan `Like` versi operator kondisi. Misalnya, alih-alih operator kondisi `StringEquals` string, gunakan`StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- Elemen kebijakan JSON IAM: Operator kondisi

- Elemen kebijakan JSON IAM: Kondisi

AWS kebijakan yang dikelola dengan peringatan umum ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan terkel AWS ola berikut menyertakan wildcard dalam nilai kondisinya tanpa operator kondisi yang disertakan `Like` untuk pencocokan pola. Saat menggunakan kebijakan ter AWS kelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, sar AWS ankan Anda menggunakan operator kondisi yang mendukung pencocokan pola dengan wildcard (*,?) , seperti`StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Peringatan Umum — Ukuran kebijakan melebihi kuota kebijakan identitas

Kode masalah: POLICY_SIZE_EXCEEDS_IDENTITY_POLICY_QUOTA

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Menyelesaikan peringatan umum

Anda dapat melampirkan hingga 20 kebijakan terkelola ke identitas IAM (pengguna, grup pengguna, atau peran). Namun, ukuran setiap kebijakan terkelola tidak boleh melebihi kuota default 6.144

karakter. IAM tidak menghitung spasi kosong saat menghitung ukuran kebijakan terhadap kuota ini. Kuota, juga disebut sebagai batas di AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di AWS akun Anda.

Selain itu, Anda dapat menambahkan kebijakan sebaris sebanyak yang Anda inginkan ke identitas IAM. Namun, ukuran jumlah semua kebijakan sebaris per identitas tidak dapat melebihi kuota yang ditentukan.

Jika kebijakan Anda lebih besar dari kuota, Anda dapat mengatur kebijakan menjadi beberapa pernyataan dan mengelompokkan pernyataan menjadi beberapa kebijakan.

Istilah terkait

- IAM dan kuota AWS STS karakter
- Beberapa pernyataan dan beberapa kebijakan
- Kebijakan yang dikelola pelanggan IAM
- Ikhtisar kebijakan JSON
- Tata bahasa kebijakan JSON IAM

AWS kebijakan yang dikelola dengan peringatan umum ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Kebijakan ter AWS kelola berikut memberikan izin untuk tindakan di banyak AWS layanan dan melebihi ukuran kebijakan maksimum. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan terkelola, Anda harus membagi kebijakan menjadi beberapa kebijakan.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Peringatan Umum — Ukuran kebijakan melebihi kuota kebijakan sumber daya

Kode masalah: POLICY_SIZE_EXCEEDS_RESOURCE_POLICY_QUOTA

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Menyelesaikan peringatan umum

Resource-based kebijakan adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya, seperti bucket Amazon S3. Kebijakan ini memberikan izin utama yang ditentukan untuk melakukan tindakan tertentu pada sumber daya tersebut dan menentukan dalam kondisi apa hal ini berlaku. Ukuran kebijakan berbasis sumber daya tidak boleh melebihi kuota yang ditetapkan untuk sumber daya tersebut. Kuota, juga disebut sebagai batas di AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di AWS akun Anda.

Jika kebijakan Anda lebih besar dari kuota, Anda dapat mengatur kebijakan menjadi beberapa pernyataan dan mengelompokkan pernyataan menjadi beberapa kebijakan.

Istilah terkait

- Resource-based kebijakan
- Kebijakan bucket Amazon S3
- Beberapa pernyataan dan beberapa kebijakan
- Ikhtisar kebijakan JSON
- Tata bahasa kebijakan JSON IAM

Peringatan Umum - Jenis ketidakcocokan

Kode masalah: TYPE_MISMATCH

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi yang didukung.

Misalnya, kunci kondisi `aws:MultiFactorAuthPresent` global memerlukan operator kondisi dengan tipe `Boolean` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Kunci kondisi global
- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Umum - Jenis ketidakcocokan Boolean

Kode masalah: TYPE_MISMATCH_BOOLEAN

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi Boolean, seperti `true` atau. `false`

Misalnya, kunci kondisi `aws:MultiFactorAuthPresent` global memerlukan operator kondisi dengan tipe `Boolean` data. Jika Anda memberikan tanggal atau bilangan bulat, tipe data tidak akan cocok.

Istilah terkait

- Operator kondisi Boolean
- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Umum - Jenis tanggal ketidakcocokan

Kode masalah: TYPE_MISMATCH_DATE

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi tanggal, dalam format waktu tanggal ISO 8601 `YYYY-MM-DD` atau lainnya.

Istilah terkait

- Operator kondisi tanggal
- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Umum - Jenis nomor ketidakcocokan

Kode masalah: TYPE_MISMATCH_NUMBER

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi numerik.

Istilah terkait

- Operator kondisi numerik
- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Umum - Jenis string ketidakcocokan

Kode masalah: TYPE_MISMATCH_STRING

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Menyelesaikan peringatan umum

Perbarui teks untuk menggunakan tipe data operator kondisi string.

Istilah terkait

- Operator kondisi string
- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Umum - Rekomendasi repo dan cabang github tertentu

Kode masalah: SPECIFIC_GITHUB_REPO_AND_BRANCH_RECOMMENDED

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name. If the subject claim does not include the branch, specify
 it in a token.actions.githubusercontent.com:ref condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value
 of token.actions.githubusercontent.com:sub with the repository and branch
 name. If the subject claim does not include the branch, specify it in a
 token.actions.githubusercontent.com:ref condition key."
```

Menyelesaikan peringatan umum

Jika Anda menggunakan GitHub sebagai IDP OIDC, praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IdP IAM. Bila Anda menyertakan `Condition` pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan kunci kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi ke kumpulan repositori atau cabang tertentu. Jika Anda menggunakan wildcard (`*`) di`token.actions.githubusercontent.com:sub`, Tindakan GitHub dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan IdP GitHub IAM di akun Anda. AWS

Tentukan repositori di`token.actions.githubusercontent.com:sub`.
Anda dapat menentukan cabang dalam nilai yang sama, atau dalam kunci `token.actions.githubusercontent.com:ref` kondisi terpisah. Ketika klaim subjek tidak menyertakan cabang - misalnya, ketika memberi nama GitHub lingkungan - tentukan cabang di`token.actions.githubusercontent.com:ref`.

Istilah terkait

- Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC

Peringatan Umum — Ukuran kebijakan melebihi kuota kebijakan kepercayaan peran

Kode masalah: POLICY_SIZE_EXCEEDS_ROLE_TRUST_POLICY_QUOTA

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Menyelesaikan peringatan umum

IAM dan AWS STS memiliki kuota yang membatasi ukuran kebijakan kepercayaan peran. Karakter dalam kebijakan kepercayaan peran, tidak termasuk spasi, melebihi maksimum karakter. Sebaiknya Anda meminta peningkatan kuota panjang kebijakan kepercayaan peran menggunakan Kuota Layanan dan AWS Support Center Console.

Istilah terkait

- IAM dan ku AWS STS ota, persyaratan nama, dan batas karakter

Peringatan Umum - RCP kehilangan kunci kondisi utama terkait

Kode masalah: RCP_MISSING_RELATED_PRINCIPAL_CONDITION_KEY

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP missing related principal condition key: RCPs impact IAM roles, users, and AWS
 service principals. To prevent unintended impact to services acting on your behalf
 using a service principal, an additional statement should be added to the Condition
 block "BoolIfExists": { "aws:PrincipalIsAWSService": "false"} whenever a principal key
 {{conditionKeyName}} is used.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to services acting on your behalf using a service principal,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "false"} whenever a principal key {{conditionKeyName}}
 is used."
```

Menyelesaikan peringatan umum

AWS Organizations kebijakan kontrol sumber daya (RCP) dapat memengaruhi peran IAM, pengguna, dan Layanan AWS prinsipal. Untuk mencegah dampak yang tidak diinginkan terhadap layanan yang bertindak atas nama Anda menggunakan prinsip layanan, tambahkan pernyataan berikut ke `Condition` elemen Anda:

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "false"}
```

Istilah terkait

- Sintaks RCP
- Properti prinsipal

Peringatan Umum - RCP kehilangan kunci kondisi utama layanan terkait

Kode masalah: RCP_MISSING_RELATED_SERVICE_PRINCIPAL_CONDITION_KEY

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP missing related service principal condition key: RCPs impact IAM roles, users,
 and AWS service principals. To prevent unintended impact to your principals,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "true"} whenever the key {{conditionKeyName}} is used.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to your principals, an additional statement should be added to the
 Condition block "BoolIfExists": { "aws:PrincipalIsAWSService": "true"} whenever the
 key {{conditionKeyName}} is used."
```

Menyelesaikan peringatan umum

AWS Organizations kebijakan kontrol sumber daya (RCP) dapat memengaruhi peran IAM, pengguna, dan Layanan AWS prinsipal. Untuk mencegah dampak yang tidak diinginkan pada prinsipal Anda, tambahkan pernyataan berikut ke elemen Anda: `Condition`

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "true"}
```

Istilah terkait

- Sintaks RCP
- Properti prinsipal

Peringatan Umum - Pemeriksaan nol kunci kondisi layanan hilang RCP

Kode masalah: RCP_MISSING_SERVICE_CONDITION_KEY_NULL_CHECK

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
RCP missing service condition key null check: The specified service may have a service
 integration that does not require the use of the {{conditionKeyName}} condition
 key. To prevent unintended impact to services acting on your behalf using a service
 principal, an additional statement should be added to the Condition block "Null":
 { "aws:SourceAccount": "false"} or "Null": { "aws:SourceArn": "false"} whenever the
 key {{conditionKeyName}} is used.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The specified service may have a service integration that does
 not require the use of the {{conditionKeyName}} condition key. To prevent unintended
```

```
impact to services acting on your behalf using a service principal, an additional
statement should be added to the Condition block "Null": { "aws:SourceAccount":
"false"} or "Null": { "aws:SourceArn": "false"} whenever the key {{conditionKeyName}}
is used."
```

Menyelesaikan peringatan umum

AWS Organizations kebijakan kontrol sumber daya (RCP) dapat memengaruhi peran IAM, pengguna, dan Layanan AWS prinsipal. Untuk mencegah dampak yang tidak diinginkan terhadap layanan yang bertindak atas nama Anda menggunakan prinsip layanan, tambahkan salah satu pernyataan berikut ke `Condition` elemen Anda setiap kali kunci yang ditentukan digunakan:

```
"Null": { "aws:SourceAccount": "false"}
```

atau

```
"Null": { "aws:SourceArn": "false"}
```

Istilah terkait

- Sintaks RCP
- Properti prinsipal

Peringatan Umum — Gunakan kunci kondisi hanya dengan layanan yang didukung

Kode masalah: USE_CONDITION_KEY_ONLY_WITH_SUPPORTED_SERVICES

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Use condition key only with supported services: The condition key {{key}} works only
 with specific AWS services and must be scoped to supported services in your policies.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{key}} works only with specific AWS services and
 must be scoped to supported services in your policies.
```

Menyelesaikan peringatan umum

AWS Tinjau dokumentasi untuk mengidentifikasi yang Layanan AWS mendukung kunci kondisi ini. Jika ada layanan dalam kebijakan Anda yang tidak mendukung kunci kondisi, ubah kebijakan Anda untuk mencakup kunci kondisi hanya untuk Layanan AWS yang mendukungnya.

Istilah terkait

- `aws:VpceAccount`
- `aws:VpceOrgID`
- `aws:VpceOrgPaths`
- Tindakan, sumber daya, dan kunci kondisi untuk Layanan AWS

Peringatan Umum — DynamoDB mengganti operasi dengan atribut

Kode masalah: DYNAMODB_REPLACE_OPERATION_WITH_ATTRIBUTES

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB replace operation with attributes: The {{action}} action replaces the entire
 item, so dynamodb:Attributes cannot restrict access at the attribute level. That
 condition only applies to UpdateItem; use dynamodb:UpdateItem for attribute-level
 access control.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{action}} action replaces the entire item, so
 dynamodb:Attributes cannot restrict access at the attribute level. That condition only
 applies to UpdateItem; use dynamodb:UpdateItem for attribute-level access control."
```

Menyelesaikan peringatan umum

`dynamodb:Attributes`Kondisi ini tidak membatasi `dynamodb:PutItem` atau`dynamodb:BatchWriteItem`. Tindakan ini menggantikan seluruh item, sehingga pemanggil dapat menghapus atribut terbatas hanya dengan meninggalkannya dari item yang mereka tulis. Untuk mengontrol atribut mana yang dapat diubah oleh pemanggil, berikan sebagai `dynamodb:UpdateItem` gantinya dan pertahankan `dynamodb:Attributes` kondisi pada pernyataan itu. Jika Anda tidak bermaksud mengontrol tingkat atribut di sini, hapus kondisi tersebut`dynamodb:Attributes`.

Peringatan Umum — Pemindaian DynamoDB melewati kunci utama

Kode masalah: DYNAMODB_SCAN_BYPASSES_LEADING_KEYS

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB scan bypasses leading keys: The dynamodb:Scan operation returns all items
 in the table, bypassing the dynamodb:LeadingKeys restriction. Consider removing
 dynamodb:Scan.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The dynamodb:Scan operation returns all items in the table,
 bypassing the dynamodb:LeadingKeys restriction. Consider removing dynamodb:Scan."
```

Menyelesaikan peringatan umum

`dynamodb:Scan`membaca seluruh tabel, sehingga `dynamodb:LeadingKeys` kondisi dalam pernyataan ini tidak membatasi item mana yang dapat dibaca oleh pemanggil. Hapus`dynamodb:Scan`, atau berikan dalam pernyataan terpisah yang tidak bergantung pada`dynamodb:LeadingKeys`.

Peringatan Umum — Kondisi DynamoDB salah untuk tindakan

Kode masalah: DYNAMODB_WRONG_CONDITION_FOR_ACTION

Jenis pencarian: GENERAL_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB wrong condition for action: The condition key {{conditionKey}} does
 not apply to the {{action}} action. Use dynamodb:Select with read actions and
 dynamodb:ReturnValues with write actions.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{conditionKey}} does not apply to the {{action}}
 action. Use dynamodb:Select with read actions and dynamodb:ReturnValues with write
 actions."
```

Menyelesaikan peringatan umum

Pernyataan ini `dynamodb:Select` digunakan dengan tindakan tulis, atau `dynamodb:ReturnValues` dengan tindakan baca. DynamoDB `dynamodb:Select` hanya menyediakan untuk tindakan baca dan `dynamodb:ReturnValues` hanya untuk tindakan tulis, jadi kondisi di sini tidak pernah berlaku dan tidak melindungi apa pun. Pasangkan setiap tombol dengan tindakan pendukung, atau hapus. Pemeriksaan ini berlaku untuk keduanya `Allow` dan `Deny` pernyataan.

Peringatan Keamanan — Kunci kondisi yang tidak dapat dipercaya

Kode masalah: UNTRUSTWORTHY_CONDITION_KEY

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Untrustworthy condition key: The {{key}} condition key is not recommended for access
 control as it can be spoofed/manipulated by the caller.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{key}} condition key is not recommended for access control as
 it can be spoofed/manipulated by the caller."
```

Menyelesaikan peringatan keamanan

Jangan gunakan kunci kondisi ini untuk kontrol akses. Penelepon berpotensi memanipulasi atau memalsukan nilai kunci, yang menciptakan risiko keamanan.

Istilah terkait

- AWS kunci konteks kondisi global

Peringatan Keamanan - Izinkan dengan NotPrincipal

Kode masalah: ALLOW_WITH_NOT_PRINCIPAL

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Menyelesaikan peringatan keamanan

Menggunakan `"Effect": "Allow"` dengan `NotPrincipal` bisa terlalu permisif. Misalnya, ini dapat memberikan izin kepada prinsipal anonim. AWS merekomendasikan agar Anda menentukan prinsipal yang memerlukan akses menggunakan `Principal` elemen. Atau, Anda dapat mengizinkan akses luas dan kemudian menambahkan pernyataan lain yang menggunakan `NotPrincipal` elemen dengan`“Effect”: “Deny”`.

- AWS Elemen kebijakan JSON: Principal
- AWS Elemen kebijakan JSON: NotPrincipal

Peringatan Keamanan - ForAllValues dengan kunci bernilai tunggal

Kode masalah: FORALLVALUES_WITH_SINGLE_VALUED_KEY

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Menyelesaikan peringatan keamanan

AWS merekomendasikan agar Anda menggunakan `ForAllValues` satu-satunya dengan kondisi multinilai. Operator `ForAllValues` set menguji apakah nilai setiap anggota kumpulan permintaan adalah subset dari kumpulan kunci kondisi. Kondisi tersebut akan memberikan hasil benar jika setiap nilai kunci dalam permintaan tersebut sesuai dengan setidaknya satu nilai dalam kebijakan. Kondisi ini juga akan memberikan hasil benar jika tidak ada kunci dalam permintaan, atau jika nilai kunci menghasilkan kumpulan data nol, seperti string kosong.

Untuk mempelajari apakah suatu kondisi mendukung nilai tunggal atau beberapa nilai, tinjau halaman Tindakan, sumber daya, dan kunci kondisi untuk layanan. Kunci kondisi dengan awalan tipe `ArrayOf` data adalah kunci kondisi multivalue. Misalnya, Amazon SES mendukung kunci dengan nilai tunggal (`String`) dan tipe data `ArrayOfString` multivalue.

- Kunci konteks multinilai

Peringatan Keamanan - Lulus peran dengan NotResource

Kode masalah: PASS_ROLE_WITH_NOT_RESOURCE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Menggunakan `iam:PassRole` dalam kebijakan dengan `NotResource` elemen dapat memungkinkan kepala sekolah mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

- Meneruskan peran ke layanan
- sudah: PassedToService
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Keamanan — Lulus peran dengan bintang beraksi dan NotResource

Kode masalah: PASS_ROLE_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) di `Action` dan yang menyertakan `NotResource` elemen dapat memungkinkan kepala sekolah Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

- Meneruskan peran ke layanan
- sudah: PassedToService
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Keamanan — Lulus peran dengan NotAction dan NotResource

Kode masalah: PASS_ROLE_WITH_NOT_ACTION_AND_NOT_RESOURCE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Menggunakan `NotAction` elemen dan mencantumkan beberapa sumber daya dalam `NotResource` elemen dapat memungkinkan kepala sekolah Anda mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

- Meneruskan peran ke layanan
- sudah: PassedToService
- Elemen kebijakan JSON IAM: NotAction
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: NotResource
- Elemen kebijakan JSON IAM: Sumber

Peringatan Keamanan — Lulus peran dengan bintang di sumber daya

Kode masalah: PASS_ROLE_WITH_STAR_IN_RESOURCE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Kebijakan yang mengi `iam:PassRole` zinkan dan menyertakan wildcard (*) dalam `Resource` elemen dapat memungkinkan kepala sekolah mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

Beberapa AWS layanan menyertakan namespace layanan mereka atas nama peran mereka. Pemeriksaan kebijakan ini memperhitungkan konvensi ini saat menganalisis kebijakan untuk menghasilkan temuan. Misalnya, sumber daya ARN berikut mungkin tidak menghasilkan temuan:

```
arn:aws:iam::*:role/Service*
```

- Meneruskan peran ke layanan
- sudah: PassedToService
- Elemen kebijakan JSON IAM: Sumber daya

AWS kebijakan yang dikelola dengan peringatan keamanan ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Salah satu kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan terkel AWS ola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan peran IAM apa pun ke layanan apa pun. AWS menyarankan agar Anda melampirkan kebijakan ter AWS kelola berikut hanya ke identitas IAM yang Anda anggap administrator.

- AdministratorAccess-Amplify

Kebijakan ter AWS kelola berikut menyertakan izin ke `iam:PassRole` dengan wildcard (*) di sumber daya dan berada di jalur penghenti an. Untuk setiap kebijakan ini, kami memperbarui panduan izin, seperti merekomendasikan kebijakan terkel AWS ola baru yang mendukung kasus penggunaan. Untuk melihat alternatif kebijakan ini, lihat panduan untuk setiap layanan.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonEC2ContainerServiceFullAccess

Kebijakan ter AWS kelola berikut memberikan izin hanya untuk peran terkait layanan, yang memungkinkan AWS layanan melakukan tindakan atas nama Anda. Anda tidak dapat melampirkan kebijakan ini ke identitas IAM Anda.

- AWSServiceRoleForAmazonEKSNodegroup

Peringatan Keamanan — Lulus peran dengan bintang dalam aksi dan sumber daya

Kode masalah: PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Kebijakan dengan wildcard (*) di `Resource` elemen `Action` dan dapat memungkinkan kepala sekolah mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

- Meneruskan peran ke layanan
- sudah: PassedToService
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: Sumber

AWS kebijakan yang dikelola dengan peringatan keamanan ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Beberapa kasus penggunaan tersebut adalah untuk administrator dalam akun Anda. Kebijakan terkel AWS ola berikut menyediakan akses administrator dan memberikan izin untuk meneruskan peran

IAM apa pun ke AWS layanan apa pun. AWS menyarankan agar Anda melampirkan kebijakan ter AWS kelola berikut hanya ke identitas IAM yang Anda anggap administrator.

- AdministratorAccess
- IAMFullAccess

Peringatan Keamanan — Lulus peran dengan bintang di sumber daya dan NotAction

Kode masalah: PASS_ROLE_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Menyelesaikan peringatan keamanan

Untuk mengkonfigurasi banyak AWS layanan, Anda harus meneruskan peran IAM ke layanan. Untuk mengizinkan ini, Anda harus memberikan `iam:PassRole` izin ke identitas (pengguna, grup pengguna, atau peran). Menggunakan `NotAction` elemen dalam kebijakan dengan wildcard (*) di `Resource` elemen dapat memungkinkan kepala sekolah mengakses lebih banyak layanan atau fitur daripada yang Anda inginkan. AWS merekomendasikan agar Anda menentukan ARN yang diizinkan di `Resource` elemen sebagai gantinya. Selain itu, Anda dapat mengurangi izin ke satu layanan dengan menggunakan kunci `iam:PassedToService` kondisi.

- Meneruskan peran ke layanan

- sudah: PassedToService
- Elemen kebijakan JSON IAM: NotAction
- Elemen kebijakan JSON IAM: Tindakan
- Elemen kebijakan JSON IAM: Sumber

Peringatan Keamanan - Kunci kondisi yang dipasangkan hilang

Kode masalah: MISSING_PAIRED_CONDITION_KEYS

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Menyelesaikan peringatan keamanan

Beberapa kunci kondisi lebih aman saat dipasangkan dengan kunci kondisi terkait lainnya. AWS merekomendasikan agar Anda menyertakan kunci kondisi terkait dalam blok kondisi yang sama dengan kunci kondisi yang ada. Ini membuat izin yang diberikan melalui kebijakan lebih aman.

Misalnya, Anda dapat menggunakan kunci `aws:VpcSourceIp` kondisi untuk membandingkan alamat IP dari mana permintaan dibuat dengan alamat IP yang Anda tentukan dalam kebijakan. AWS menyarankan agar Anda menambahkan kunci `aws:SourceVPC` kondisi terkait. Ini memeriksa

apakah permintaan berasal dari VPC yang Anda tentukan dalam kebijakan dan alamat IP yang Anda tentukan.

Istilah terkait

- `aws:VpcSourceIp`kunci kondisi global
- `aws:SourceVPC`kunci kondisi global
- Kunci kondisi global
- Elemen kondisi
- Ikhtisar kebijakan JSON

Peringatan Keamanan — Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan

Kode masalah: DENY_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Menyelesaikan peringatan keamanan

Menggunakan kunci kondisi tag yang tidak didukung dalam `Condition` elemen kebijakan dengan `"Effect": "Deny"` bisa terlalu permisif, karena kondisi tersebut diabaikan untuk layanan tersebut. AWS menyarankan agar Anda menghapus tindakan layanan yang tidak mendukung kunci kondisi dan membuat pernyataan lain untuk menolak akses ke sumber daya tertentu untuk tindakan tersebut.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dan tidak didukung oleh tindakan layanan, maka kunci tersebut tidak disertakan dalam konteks permintaan. Dalam hal ini, kondisi dalam `Deny` pernyataan selalu kembali `false` dan tindakan tidak pernah ditolak. Ini terjadi bahkan jika sumber daya ditandai dengan benar.

Bila layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut (ABAC). Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (RBAC).

> **Note**
>
> Beberapa layanan memungkinkan dukungan untuk kunci `aws:ResourceTag` kondisi untuk subset sumber daya dan tindakan mereka. IAM Access Analyzer mengembalikan temuan untuk tindakan layanan yang tidak didukung. Misalnya, Amazon S3 mendukung sub `aws:ResourceTag` set sumber dayanya. Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3 yang mendukung kunci `aws:ResourceTag` kondisi, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Misalnya, asumsikan bahwa Anda ingin menolak akses untuk menghapus tag menghapus sumber daya tertentu yang ditandai dengan pasangan nilai kunci. `status=Confidential` Juga asumsikan itu AWS Lambda memungkinkan Anda untuk menandai dan menghapus tag sumber daya, tetapi tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk menolak tindakan hapus untuk AWS App Mesh dan AWS Backup jika tag ini ada, gunakan tombol `aws:ResourceTag` kondisi. Untuk Lambda, gunakan konvensi penamaan sumber daya yang menyertakan `"Confidential"` awalan. Kemudian sertakan pernyataan terpisah yang mencegah penghapusan sumber daya dengan konvensi penamaan tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> Jangan gunakan IfExists versi... dari operator kondisi sebagai solusi untuk temuan ini. Ini berarti “Tolak tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Kalau tidak, tolak tindakan itu.” Dalam contoh sebelumnya, termasuk `lambda:DeleteFunction` tindakan dalam `DenyDeleteSupported` pernyataan dengan `StringEqualsIfExists` operator selalu menyangkal tindakan. Untuk tindakan itu, kunci tidak ada dalam konteks, dan setiap upaya untuk menghapus jenis sumber daya tersebut ditolak, terlepas dari apakah sumber daya tersebut diberi tag.

Istilah terkait

- Kunci kondisi global
- Membandingkan ABAC ke RBAC
- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kondisi
- Ikhtisar kebijakan JSON

Peringatan Keamanan — Tolak NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Kode masalah:
DENY_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Menyelesaikan peringatan keamanan

Menggunakan kunci kondisi tag dalam `Condition` elemen kebijakan dengan elemen `NotAction` dan `"Effect": "Deny"` bisa terlalu permisif. Kondisi diabaikan untuk tindakan layanan yang tidak mendukung kunci kondisi. AWS merekomendasikan agar Anda menulis ulang logika untuk menolak daftar tindakan.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dengan`NotAction`, tindakan layanan baru atau yang sudah ada yang tidak mendukung kunci tidak ditolak. AWS menyarankan agar Anda secara eksplisit mencantumkan tindakan yang ingin Anda tolak. IAM Access Analyzer mengembalikan temuan terpisah untuk tindakan terdaftar yang tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk informasi selengkapnya, lihat Peringatan Keamanan — Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan.

Bila layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut (ABAC). Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (RBAC).

Istilah terkait

- Kunci kondisi global
- Membandingkan ABAC ke RBAC
- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kondisi
- Ikhtisar kebijakan JSON

Peringatan Keamanan — Batasi akses ke prinsipal layanan

Kode masalah: RESTRICT_ACCESS_TO_SERVICE_PRINCIPAL

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Menyelesaikan peringatan keamanan

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengidentifikasi untuk layanan. Saat memberikan akses ke prinsipal layanan untuk bertindak atas nama Anda, batasi

akses. Anda dapat mencegah kebijakan yang terlalu permisif dengan menggunakan kunci `aws:SourceArnaws:SourceAccount`,`aws:SourceOrgID`,, atau `aws:SourceOrgPaths` kondisi untuk membatasi akses ke sumber tertentu, seperti ARN sumber daya tertentu Akun AWS, ID organisasi, atau jalur organisasi. Membatasi akses membantu Anda mencegah masalah keamanan yang disebut masalah wakil yang membingungkan.

Istilah terkait

- Layanan AWS kepala sekolah
- AWS kunci kondisi global: aws: SourceAccount
- AWS kunci kondisi global: aws: SourceArn
- AWS kunci kondisi global: aws: SourceOrgId
- AWS kunci kondisi global: aws: SourceOrgPaths
- Masalah wakil yang membingungkan

Peringatan Keamanan - Kunci kondisi hilang untuk prinsipal oidc

Kode masalah: MISSING_CONDITION_KEY_FOR_OIDC_PRINCIPAL

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Menyelesaikan peringatan keamanan

Menggunakan prinsip Open ID Connect tanpa syarat bisa terlalu permisif. Tambahkan kunci kondisi dengan awalan yang cocok dengan prinsipal OIDC federasi Anda untuk memastikan bahwa hanya penyedia identitas yang dituju yang mengambil peran tersebut.

Istilah terkait

- Membuat peran untuk identitas web atau OpenID Connect Federation (konsol)

Peringatan Keamanan - Kunci kondisi repo github hilang

Kode masalah: MISSING_GITHUB_REPO_CONDITION_KEY

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Menyelesaikan peringatan keamanan

Jika Anda menggunakan GitHub sebagai IDP OIDC, praktik terbaik adalah membatasi entitas yang dapat mengambil peran yang terkait dengan IdP IAM. Bila Anda menyertakan `Condition` pernyataan dalam kebijakan kepercayaan peran, Anda dapat membatasi peran tersebut ke GitHub organisasi, repositori, atau cabang tertentu. Anda dapat menggunakan kunci kondisi `token.actions.githubusercontent.com:sub` untuk membatasi akses. Kami menyarankan Anda membatasi kondisi ke kumpulan repositori atau cabang tertentu. Jika Anda tidak menyertakan

kondisi ini, Tind GitHub akan dari organisasi atau repositori di luar kendali Anda dapat mengambil peran yang terkait dengan IdP GitHub IAM di akun Anda. AWS

Istilah terkait

- Mengkonfigurasi peran untuk penyedia identitas GitHub OIDC

Peringatan Keamanan - String seperti operator dengan kunci kondisi ARN

Kode masalah: STRING_LIKE_OPERATOR_WITH_ARN_CONDITION_KEYS

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
String like operator with ARN condition keys: Use the operator type {{allowed}} instead
 of operator {{operator}} for the condition key {{key}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Menyelesaikan peringatan keamanan

AWS merekomendasikan agar Anda menggunakan operator ARN alih-alih operator string saat membandingkan ARN untuk memastikan pembatasan akses yang tepat berdasarkan nilai kondisi ARN. Perbarui `StringLike` operator ke `ArnLike` operator di `Condition` elemen Anda setiap kali kunci yang ditentukan digunakan.

Kebijakan ter AWS kelola ini merupakan pengecualian untuk peringatan keamanan ini:

- AmazonSecurityLakeAdministrator
- AWSCodePipeline_FullAccess
- AWSCodePipeline_ReadOnlyAccess
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Istilah terkait

- Operator kondisi Nama Sumber Daya Amazon (ARN)
- Operator kondisi string
- AWS kebijakan yang dikelola

Peringatan Keamanan — atribut DynamoDB tanpa pilih

Kode masalah: DYNAMODB_ATTRIBUTES_WITHOUT_SELECT

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB attributes without select: Restricting dynamodb:Attributes without also
 setting dynamodb:Select to SPECIFIC_ATTRIBUTES allows all attributes to be returned
 on read requests that omit a projection expression. We recommend that you also set
 dynamodb:Select to SPECIFIC_ATTRIBUTES.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Restricting dynamodb:Attributes without also setting dynamodb:Select
 to SPECIFIC_ATTRIBUTES allows all attributes to be returned on read requests that
 omit a projection expression. We recommend that you also set dynamodb:Select to
 SPECIFIC_ATTRIBUTES."
```

Menyelesaikan peringatan keamanan

Tambahkan `StringEquals` kondisi yang ditetapkan `dynamodb:Select` ke `SPECIFIC_ATTRIBUTES` dalam pernyataan yang sama dengan `dynamodb:Attributes` kondisi. DynamoDB mengevaluasi `dynamodb:Attributes` hanya untuk permintaan yang menentukan atribut yang akan dikembalikan. Tanpa kondisi ini, permintaan baca yang menghilangkan ekspresi proyeksi mengembalikan seluruh item.

Istilah terkait

- Menggunakan kondisi kebijakan IAM untuk kontrol akses yang halus

- Elemen kebijakan JSON IAM: Operator kondisi

Peringatan Keamanan - Awalan S3 dalam konteks negatif

Kode masalah: S3_PREFIX_IN_NEGATIVE_CONTEXT

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
S3 prefix in negative context: Using the s3:prefix condition key in a Deny statement or
 with a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the s3:prefix condition key in a Deny statement or with
 a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike."
```

Menyelesaikan peringatan keamanan

Untuk membatasi daftar pada awalan tertentu, gunakan salah satu pola berikut:

- Izinkan dengan StringLike — B `s3:ListBucket` eri dengan `StringLike` syarat untuk `s3:prefix` mencantumkan awalan yang ingin Anda izinkan. Permintaan yang menyediakan awalan yang berbeda tidak tercakup oleh Izinkan ini dan tidak dapat mencantumkan objek tersebut.
- Tolak dengan StringNotLike — Tol `s3:ListBucket` ak dengan `StringNotLike` syarat untuk `s3:prefix` mencantumkan awalan yang ingin Anda izinkan. Permintaan yang menghilangkan awalan atau menyediakan awalan yang lebih pendek tidak cocok dengan set yang diizinkan dan ditolak.

Sebuah `Deny` dengan `StringLike` atau `Allow` dengan `StringNotLike` tidak secara efektif membatasi akses karena pemanggil dapat melewati kondisi dengan menentukan awalan yang lebih pendek (misalnya, `priv` bukan`private/`) yang tidak cocok dengan nilai kondisi.

Istilah terkait

- Contoh kunci kondisi Amazon S3
- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kebijakan IAM: Kondisi

Peringatan Keamanan — ForAnyValue dengan jenis klaim audiens

Kode masalah: FORANYVALUE_WITH_AUDIENCE_CLAIM_TYPE

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
ForAnyValue with audience claim type: Using ForAnyValue qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAnyValue:.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using ForAnyValue qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAnyValue:."
```

Menyelesaikan peringatan keamanan

AWS merekomendasikan agar Anda tidak menggunakan operator `ForAnyValue` set dengan kunci kondisi bernilai tunggal. Gunakan operator set hanya dengan kunci kondisi multivalue. Hapus operator `ForAnyValue` set.

Istilah terkait

- Single-valued vs. kunci konteks multinilai
- Single-valued contoh kebijakan utama konteks

Peringatan Keamanan — atribut DynamoDB tanpa nilai pengembalian

Kode masalah: DYNAMODB_ATTRIBUTES_WITHOUT_RETURN_VALUES

Jenis pencarian: SECURITY_WARNING

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
DynamoDB attributes without return values: Restricting dynamodb:Attributes without
 limiting dynamodb:ReturnValues allows write operations to return the entire item,
 which can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Restricting dynamodb:Attributes without limiting
 dynamodb:ReturnValues allows write operations to return the entire item, which
 can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW."
```

Menyelesaikan peringatan keamanan

Tambahkan kondisi yang membatasi `dynamodb:ReturnValues` ke`NONE`,`UPDATED_OLD`, atau `UPDATED_NEW` dalam pernyataan yang sama dengan `dynamodb:Attributes` kondisi. Tanpa batas ini, tulisan yang `dynamodb:ReturnValues` menyet `ALL_OLD` el atau `ALL_NEW` mengembalikan seluruh item, termasuk atribut di luar yang Anda izinkan.

Saran - Tindakan array kosong

Kode masalah: EMPTY_ARRAY_ACTION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Menyelesaikan saran

Pernyataan harus mencakup `NotAction` elemen `Action` atau yang mencakup serangkaian tindakan. Saat elemen kosong, pernyataan kebijakan tidak memberikan izin. Tentukan tindakan dalam `Action` elemen.

- Elemen kebijakan JSON IAM: Tindakan

Saran - Kondisi array kosong

Kode masalah: EMPTY_ARRAY_CONDITION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen opsional mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Ketika nilai kondisi kosong, kondisi akan kembali `true` dan pernyataan kebijakan tidak memberikan izin. Tentukan nilai kondisi.

- Elemen kebijakan JSON IAM: Kondisi

Saran - Kondisi array kosong ForAllValues

Kode masalah: EMPTY_ARRAY_CONDITION_FORALLVALUES

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Operator `ForAllValues` set menguji apakah nilai setiap anggota kumpulan permintaan adalah subset dari kumpulan kunci kondisi.

Saat Anda menggunakan `ForAllValues` dengan kunci kondisi kosong, kondisi hanya cocok jika tidak ada kunci dalam permintaan. AWS merekomendasikan bahwa jika Anda ingin menguji apakah konteks permintaan kosong, gunakan operator `Null` kondisi sebagai gantinya.

- Kunci konteks multinilai
- Operator kondisi nol
- Elemen kebijakan JSON IAM: Kondisi

Saran - Kondisi array kosong ForAnyValue

Kode masalah: EMPTY_ARRAY_CONDITION_FORANYVALUE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai. Operator `ForAnyValues` set menguji apakah setidaknya satu anggota dari himpunan nilai permintaan cocok dengan setidaknya satu anggota dari himpunan nilai kunci kondisi.

Ketika Anda menggunakan `ForAnyValues` dengan kunci kondisi kosong, kondisi tidak pernah cocok. Ini berarti bahwa pernyataan tersebut tidak berpengaruh pada kebijakan. AWS menyarankan agar Anda menulis ulang kondisi.

- Kunci konteks multinilai
- Elemen kebijakan JSON IAM: Kondisi

Saran - Kondisi array kosong IfExists

Kode masalah: EMPTY_ARRAY_CONDITION_IFEXISTS

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
```

```
request context is empty, we recommend that you use the Null condition operator with
the value of true instead.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Menyelesaikan saran

Su `...IfExists` fiks mengedit operator kondisi. Ini berarti bahwa jika kunci kebijakan hadir dalam konteks permintaan, proses kunci seperti yang ditentukan dalam kebijakan. Jika kunci tidak ada, evaluasi elemen kondisi sebagai benar.

Saat Anda menggunakan `...IfExists` dengan kunci kondisi kosong, kondisi hanya cocok jika tidak ada kunci dalam permintaan. AWS merekomendasikan bahwa jika Anda ingin menguji apakah konteks permintaan kosong, gunakan operator `Null` kondisi sebagai gantinya.

- ... IfExists operator kondisi
- Elemen kebijakan JSON IAM: Kondisi

Saran - Prinsip array kosong

Kode masalah: EMPTY_ARRAY_PRINCIPAL

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Menyelesaikan saran

Anda harus menggunakan `NotPrincipal` elemen `Principal` atau dalam kebijakan kepercayaan untuk peran IAM dan kebijakan berbasis sumber daya. Resource-based kebijakan adalah kebijakan yang Anda sematkan langsung di sumber daya.

Bila Anda memberikan array kosong di `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS menyarankan agar Anda menentukan prinsipal yang harus memiliki akses ke sumber daya.

- Elemen kebijakan JSON IAM: Principal
- Elemen kebijakan JSON IAM: NotPrincipal

Saran - Sumber daya array kosong

Kode masalah: EMPTY_ARRAY_RESOURCE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Menyelesaikan saran

Pernyataan harus menyertakan elemen `Resource` atau `NotResource`.

Bila Anda memberikan array kosong di elemen sumber daya pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS menyarankan agar Anda menentukan Nama Sumber Daya Amazon (ARN) untuk sumber daya.

- Elemen kebijakan JSON IAM: Sumber
- Elemen kebijakan JSON IAM: NotResource

Saran — Kondisi objek kosong

Kode masalah: EMPTY_OBJECT_CONDITION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Menyelesaikan saran

Struktur `Condition` elemen mengharuskan Anda menggunakan operator kondisi dan pasangan kunci-nilai.

Bila Anda memberikan objek kosong dalam elemen kondisi pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. Hapus elemen opsional atau tentukan kondisi.

- Elemen kebijakan JSON IAM: Kondisi

Saran — Prinsip objek kosong

Kode masalah: EMPTY_OBJECT_PRINCIPAL

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Menyelesaikan saran

Anda harus menggunakan `NotPrincipal` elemen `Principal` atau dalam kebijakan kepercayaan untuk peran IAM dan kebijakan berbasis sumber daya. Resource-based kebijakan adalah kebijakan yang Anda sematkan langsung di sumber daya.

Bila Anda memberikan objek kosong dalam `Principal` elemen pernyataan, pernyataan tersebut tidak berpengaruh pada kebijakan. AWS menyarankan agar Anda menentukan prinsipal yang harus memiliki akses ke sumber daya.

- Elemen kebijakan JSON IAM: Principal
- Elemen kebijakan JSON IAM: NotPrincipal

Saran — Nilai Sid kosong

Kode masalah: EMPTY_SID_VALUE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Menyelesaikan saran

Elemen opsional `Sid` (ID pernyataan) memungkinkan Anda memasukkan pengidentifikasi yang Anda berikan untuk pernyataan kebijakan. Anda dapat menetapkan `Sid` nilai untuk setiap pernyataan dalam array pernyataan. Jika Anda memilih untuk menggunakan `Sid` elemen, Anda harus memberikan nilai string.

Istilah terkait

- Elemen kebijakan JSON IAM: Sid

Saran — Setara dengan null false

Kode masalah: EQUIVALENT_TO_NULL_FALSE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Equivalent to null false: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition."
```

Menyelesaikan saran

Ganti kunci kondisi saat ini dengan kunci yang disarankan disetel ke`false`. Perubahan ini meningkatkan kejelasan kebijakan dan memastikan evaluasi kondisi yang lebih andal. Perbarui blok

kondisi Anda untuk digunakan sebagai `{recommendedKey}: false` pengganti kombinasi kunci-operator saat ini.

Istilah terkait

- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global

Saran — Setara dengan null true

Kode masalah: EQUIVALENT_TO_NULL_TRUE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Equivalent to null true: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition."
```

Menyelesaikan saran

Ganti kunci kondisi saat ini dengan kunci yang disarankan disetel `ketrue`. Perubahan ini meningkatkan kejelasan kebijakan dan memastikan evaluasi kondisi yang lebih andal. Perbarui blok kondisi Anda untuk digunakan sebagai `{recommendedKey}: true` pengganti kombinasi kunci-operator saat ini.

Istilah terkait

- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global

Saran - Tingkatkan jangkauan IP

Kode masalah: IMPROVE_IP_RANGE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Menyelesaikan saran

Kondisi alamat IP harus dalam format CIDR standar, seperti 203.0.113. 0/24 atau 2001:DB8:1234:5678: :/64. Bila Anda menyertakan bit bukan nol setelah bit bertopeng, bit tersebut tidak dipertimbangkan untuk kondisi tersebut. AWS menyarankan agar Anda menggunakan alamat baru yang disertakan dalam pesan.

- Operator kondisi alamat IP
- Elemen kebijakan JSON IAM: Kondisi

Saran — Null dengan kualifikasi

Kode masalah: NULL_WITH_QUALIFIER

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Menyelesaikan saran

Dalam `Condition` elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi seperti sama atau kurang dari untuk membandingkan kondisi dalam kebijakan terhadap kunci dan nilai dalam konteks permintaan. Untuk permintaan yang menyertakan beberapa nilai untuk kunci kondisi tunggal, Anda harus menggunakan operator `ForAllValues` atau `ForAnyValue` set.

Ketika Anda menggunakan operator `Null` kondisi dengan`ForAllValues`, pernyataan selalu kembali`true`. Ketika Anda menggunakan operator `Null` kondisi dengan`ForAnyValue`, pernyataan selalu kembali`false`. AWS merekomendasikan agar Anda menggunakan operator `StringLike` kondisi dengan operator set ini.

Istilah terkait

- Kunci konteks multinilai
- Operator kondisi nol
- Elemen kondisi

Saran - Subset alamat IP pribadi

Kode masalah: PRIVATE_IP_ADDRESS_SUBSET

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Menyelesaikan saran

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik.

Ketika `Condition` elemen Anda menyertakan campuran alamat IP pribadi dan publik, pernyataan tersebut mungkin tidak memiliki efek yang diinginkan. Anda dapat menentukan alamat IP pribadi menggunakan`aws:VpcSourceIP`.

> **Note**
>
> Kunci kondisi global hanya `aws:VpcSourceIP` cocok jika permintaan berasal dari alamat IP yang ditentukan dan melewati titik akhir VPC.

- aws: kunci kondisi SourceIp global
- aws: kunci kondisi VpcSourceIp global
- Operator kondisi alamat IP
- Elemen kebijakan JSON IAM: Kondisi

Saran - NotIpAddress Subset pribadi

Kode masalah: PRIVATE_NOT_IP_ADDRESS_SUBSET

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Menyelesaikan saran

Kunci kondisi global hanya `aws:SourceIp` berfungsi untuk rentang alamat IP publik.

Ketika `Condition` elemen Anda menyertakan operator `NotIpAddress` kondisi dan campuran alamat IP pribadi dan publik, pernyataan tersebut mungkin tidak memiliki efek yang diinginkan. Setiap alamat IP publik yang tidak ditentukan dalam kebijakan akan cocok. Tidak ada alamat IP pribadi yang cocok. Untuk mencapai efek ini, Anda dapat menggunakan `NotIpAddress` `aws:VpcSourceIP` dan menentukan alamat IP pribadi yang seharusnya tidak cocok.

- aws: kunci kondisi SourceIp global
- aws: kunci kondisi VpcSourceIp global
- Operator kondisi alamat IP
- Elemen kebijakan JSON IAM: Kondisi

Saran - Tindakan redundan

Kode masalah: REDUNDANT_ACTION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Menyelesaikan saran

Bila Anda menggunakan wildcard (*) dalam `Action` elemen, Anda dapat menyertakan izin berlebihan. AWS menyarankan agar Anda meninjau kebijakan Anda dan hanya menyertakan izin yang Anda butuhkan. Ini dapat membantu Anda menghapus tindakan berlebihan.

Misalnya, tindakan berikut mencakup `iam:GetCredentialReport` tindakan dua kali.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

Dalam contoh ini, izin ditentukan untuk setiap tindakan IAM yang dimulai dengan `Get` atau`List`. Saat IAM menambahkan operasi get atau list tambahan, kebijakan ini akan mengizinkannya. Anda mungkin ingin mengizinkan semua tindakan read-only ini. T `iam:GetCredentialReport` indakan sudah termasuk sebagai bagian dari`iam:Get*`. Untuk menghapus izin duplikat, Anda dapat menghapus`iam:GetCredentialReport`.

Anda menerima temuan untuk pemeriksaan kebijakan ini jika semua konten tindakan berlebihan. Dalam contoh ini, jika elemen disertakan`iam:*CredentialReport`, itu tidak dianggap berlebihan. Itu termasuk`iam:GetCredentialReport`, yang berlebihan, dan`iam:GenerateCredentialReport`, yang tidak. Menghapus `iam:Get*` salah `iam:*CredentialReport` satu atau akan mengubah izin kebijakan.

- Elemen kebijakan JSON IAM: Tindakan

AWS kebijakan yang dikelola dengan saran ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Tindakan berlebihan tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, AWS Anda disarankan untuk menghapus tindakan berlebihan dari kebijakan Anda.

Saran — Nilai kondisi redundan num

Kode masalah: REDUNDANT_CONDITION_VALUE_NUM

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Menyelesaikan saran

Bila Anda menggunakan operator kondisi numerik untuk nilai serupa dalam kunci kondisi, Anda dapat membuat tumpang tindih yang menghasilkan izin berlebihan.

Misalnya, `Condition` elemen berikut mencakup beberapa `aws:MultiFactorAuthAge` kondisi yang memiliki tumpang tindih usia 1200 detik.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
```

```
            "3600"
          ]
        }
      }
```

Dalam contoh ini, izin ditentukan jika otentikasi multi-faktor (MFA) diselesaikan kurang dari 3600 detik (1 jam) yang lalu. Anda dapat menghapus `2700` nilai berlebihan.

- Operator kondisi numerik
- Elemen kebijakan JSON IAM: Kondisi

Saran - Sumber daya redundan

Kode masalah: REDUNDANT_RESOURCE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Menyelesaikan saran

Saat menggunakan wildcard (*) di Amazon Resource Names (ARN), Anda dapat membuat izin sumber daya yang berlebihan.

Misalnya, `Resource` elemen berikut mencakup beberapa ARN dengan izin redundan.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
```

```
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

Dalam contoh ini, izin ditentukan untuk peran apa pun dengan nama yang dimulai dengan`jane`. Anda dapat menghapus redundansi `jane-admin` dan `jane-s3only` ARN tanpa mengubah izin yang dihasilkan. Hal ini membuat kebijakan menjadi dinamis. Ini akan menentukan izin untuk peran masa depan yang dimulai dengan`jane`. Jika tujuan kebijakan ini adalah untuk mengizinkan akses ke sejumlah peran statis, maka hapus ARN terakhir dan daftar hanya ARN yang harus ditentukan.

- Elemen kebijakan JSON IAM: Sumber daya

AWS kebijakan yang dikelola dengan saran ini

AWS kebijakan ter kelola memungkinkan Anda memulai AWS dengan menetapkan izin berdasarkan kasus AWS penggunaan umum.

Sumber daya yang berlebihan tidak memengaruhi izin yang diberikan oleh kebijakan. Saat menggunakan kebijakan AWS terkelola sebagai referensi untuk membuat kebijakan yang dikelola pelanggan, Anda AWS disarankan untuk menghapus sumber daya yang berlebihan dari kebijakan Anda.

Saran - Pernyataan redundan

Kode masalah: REDUNDANT_STATEMENT

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Menyelesaikan saran

Elemen `Statement` adalah elemen utama kebijakan ini. Elemen ini wajib diisi. Elemen `Statement` dapat berisi satu pernyataan atau serangkaian pernyataan individu.

Ketika Anda memasukkan pernyataan yang sama lebih dari sekali dalam kebijakan panjang, pernyataan tersebut berlebihan. Anda dapat menghapus salah satu pernyataan tanpa memengaruhi izin yang diberikan oleh kebijakan. Ketika seseorang mengedit kebijakan, mereka mungkin mengubah salah satu pernyataan tanpa memperbarui duplikatnya. Ini mungkin menghasilkan lebih banyak izin daripada yang dimaksudkan.

- Elemen kebijakan JSON IAM: Pernyataan

Saran - Wildcard dalam nama layanan

Kode masalah: WILDCARD_IN_SERVICE_NAME

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Menyelesaikan saran

Saat Anda menyertakan nama AWS layanan dalam kebijakan, disarankan agar AWS Anda tidak menyertakan wildcard (*,?). Ini mungkin menambahkan izin untuk layanan masa depan yang tidak Anda inginkan. Misalnya, ada lebih dari selusin AWS layanan dengan kata `*code*` dalam nama mereka.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- Elemen kebijakan JSON IAM: Sumber daya

Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan

Kode masalah: ALLOW_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Menyelesaikan saran

Menggunakan kunci kondisi tag yang tidak didukung di `Condition` elemen kebijakan dengan `"Effect": "Allow"` tidak memengaruhi izin yang diberikan oleh kebijakan, karena kondisi tersebut diabaikan untuk tindakan layanan tersebut. AWS menyarankan agar Anda menghapus tindakan untuk layanan yang tidak mendukung kunci kondisi dan membuat pernyataan lain untuk mengizinkan akses ke sumber daya tertentu dalam layanan tersebut.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dan tidak didukung oleh tindakan layanan, maka kunci tersebut tidak disertakan dalam konteks permintaan. Dalam hal ini, kondisi dalam `Allow` pernyataan selalu kembali `false` dan tindakan tidak pernah diizinkan. Ini terjadi bahkan jika sumber daya ditandai dengan benar.

Bila layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut (ABAC). Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (RBAC).

> **Note**
>
> Beberapa layanan memungkinkan dukungan untuk kunci `aws:ResourceTag` kondisi untuk subset sumber daya dan tindakan mereka. IAM Access Analyzer mengembalikan temuan untuk tindakan layanan yang tidak didukung. Misalnya, Amazon S3 mendukung sub `aws:ResourceTag` set sumber dayanya. Untuk melihat semua jenis sumber daya yang tersedia di Amazon S3 yang mendukung kunci `aws:ResourceTag` kondisi, lihat Jenis sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan.

Misalnya, Anda ingin mengizinkan anggota tim untuk melihat detail sumber daya tertentu yang diberi tag dengan pasangan kunci-nilai. `team=BumbleBee` Juga asumsikan itu AWS Lambda memungkinkan Anda untuk menandai sumber daya, tetapi tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk mengizinkan tindakan tampilan untuk AWS App Mesh dan AWS Backup jika tag ini ada, gunakan tombol `aws:ResourceTag` kondisi. Untuk Lambda, gunakan konvensi penamaan sumber daya yang menyertakan nama tim sebagai awalan. Kemudian sertakan pernyataan terpisah yang memungkinkan melihat sumber daya dengan konvensi penamaan tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
```

```
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

Warning

Jangan gunakan `Not` versi operator kondisi `"Effect": "Allow"` sebagai solusi untuk temuan ini. Operator kondisi ini memberikan pencocokan yang dinegasikan. Ini berarti bahwa setelah kondisi dievaluasi, hasilnya dinegasikan. Dalam contoh sebelumnya, menyertakan `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringNotEquals` operator selalu memungkinkan tindakan, terlepas dari apakah sumber daya ditandai.

Jangan gunakan `IfExists` versi... dari operator kondisi sebagai solusi untuk temuan ini. Ini berarti “Izinkan tindakan jika kunci ada dalam konteks permintaan dan nilainya cocok. Kalau tidak, izinkan tindakan.” Dalam contoh sebelumnya, memasukkan `lambda:GetFunction` tindakan dalam `AllowViewSupported` pernyataan dengan `StringEqualsIfExists` operator selalu memungkinkan tindakan. Untuk tindakan itu, kunci tidak ada dalam konteks, dan setiap upaya untuk melihat jenis sumber daya diperbolehkan, terlepas dari apakah sumber daya tersebut diberi tag.

Istilah terkait

- Kunci kondisi global
- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kondisi
- Ikhtisar kebijakan JSON

Saran - Izinkan NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan

Kode masalah:
ALLOW_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Menyelesaikan saran

Menggunakan kunci kondisi tag yang tidak didukung di `Condition` elemen kebijakan dengan elemen `NotAction` dan `"Effect": "Allow"` tidak memengaruhi izin yang diberikan oleh kebijakan. Kondisi diabaikan untuk tindakan layanan yang tidak mendukung kunci kondisi. AWS merekomendasikan agar Anda menulis ulang logika untuk mengizinkan daftar tindakan.

Jika Anda menggunakan kunci `aws:ResourceTag` kondisi dengan`NotAction`, tindakan layanan baru atau yang sudah ada yang tidak mendukung kunci tidak diizinkan. AWS menyarankan agar Anda secara eksplisit mencantumkan tindakan yang ingin Anda izinkan. IAM Access Analyzer mengembalikan temuan terpisah untuk tindakan terdaftar yang tidak mendukung kunci `aws:ResourceTag` kondisi. Untuk informasi selengkapnya, lihat Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan.

Bila layanan mendukung kunci `aws:ResourceTag` kondisi, Anda dapat menggunakan tag untuk mengontrol akses ke sumber daya layanan tersebut. Ini dikenal sebagai kontrol akses berbasis atribut (ABAC). Layanan yang tidak mendukung kunci ini mengharuskan Anda mengontrol akses ke sumber daya menggunakan kontrol akses berbasis sumber daya (RBAC).

Istilah terkait

- Kunci kondisi global

- Membandingkan ABAC ke RBAC
- Elemen kebijakan JSON IAM: Operator kondisi
- Elemen kondisi
- Ikhtisar kebijakan JSON

Saran - Kunci kondisi yang direkomendasikan untuk prinsipal layanan

Kode masalah: RECOMMENDED_CONDITION_KEY_FOR_SERVI CE_PRINCIPAL

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Menyelesaikan saran

Anda dapat menentukan Layanan AWS dalam `Principal` elemen kebijakan berbasis sumber daya menggunakan prinsip layanan, yang merupakan pengidentifikasi untuk layanan. Anda harus menggunakan kunci`aws:SourceArn`,`aws:SourceAccount`,`aws:SourceOrgID`, atau `aws:SourceOrgPaths` kondisi saat memberikan akses ke prinsipal layanan alih-alih kunci kondisi lainnya, seperti`aws:Referer`. Ini membantu Anda mencegah masalah keamanan yang disebut masalah wakil yang membingungkan.

Istilah terkait

- Layanan AWS kepala sekolah

- AWS kunci kondisi global: aws: SourceAccount
- AWS kunci kondisi global: aws: SourceArn
- AWS kunci kondisi global: aws: SourceOrgId
- AWS kunci kondisi global: aws: SourceOrgPaths
- Masalah wakil yang membingungkan

Saran — Kunci kondisi yang tidak relevan dalam kebijakan

Kode masalah: IRRELEVANT_CONDITION_KEY_IN_POLICY

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Menyelesaikan saran

Beberapa kunci kondisi tidak relevan untuk kebijakan berbasis sumber daya. Misalnya, kunci `s3:ResourceAccount` kondisi tidak relevan untuk kebijakan berbasis sumber daya yang dilampirkan ke bucket Amazon S3 atau jenis sumber daya titik akses Amazon S3.

Anda harus menggunakan kunci kondisi dalam kebijakan berbasis identitas untuk mengontrol akses ke sumber daya.

Istilah terkait

- Identity-based kebijakan dan kebijakan berbasis sumber daya

Saran - Kunci redundan karena kondisi wildcard

Kode masalah: REDUNDANT_KEY_DUE_TO_WILDCARD_IN_CONDITION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant key due to wildcard in condition: The key {{key}} in the condition block of
 {{operator}} is redundant because it is always matched. Remove this key to simplify
 the condition.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} is redundant
 because it is always matched. Remove this key to simplify the condition."
```

Menyelesaikan saran

Hapus kunci kondisi redundan dari kebijakan Anda. Kuncinya selalu cocok karena pola wildcard, sehingga tidak perlu. Sederhanakan blok kondisi Anda dengan menghapus kunci ini sambil mempertahankan izin efektif yang sama.

Istilah terkait

- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global

Saran — Prinsip redundan dalam kebijakan kepercayaan peran

Kode masalah: REDUNDANT_PRINCIPAL_IN_ROLE_TRUST_POLICY

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Menyelesaikan saran

Jika Anda menentukan prinsip peran yang diasumsikan dan peran induknya dalam `Principal` elemen kebijakan, itu tidak mengizinkan atau menolak izin yang berbeda. Misalnya, itu berlebihan jika Anda menentukan `Principal` elemen menggunakan format berikut:

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Sebaiknya hapus prinsip peran yang diasumsikan.

Istilah terkait

- Prinsip sesi peran

Saran - Pernyataan berlebihan karena wildcard dalam kondisi

Kode masalah: REDUNDANT_STATEMENT_DUE_TO_WILDCARD_IN_CONDITION

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Redundant statement due to wildcard in condition: The key {{key}} in the condition
 block of {{operator}} does not match any values. Remove this key to simplify the
 condition.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} does not
 match any values. Remove this key to simplify the condition."
```

Menyelesaikan saran

Hapus kunci kondisi yang tidak cocok dengan nilai apa pun. Kunci ini menciptakan kondisi yang tidak dapat dijangkau yang tidak akan pernah terpenuhi, membuatnya berlebihan. Bersihkan kebijakan Anda dengan menghapus kunci ini untuk meningkatkan keterbacaan dan kinerja.

Istilah terkait

- Elemen kebijakan IAM: Kondisi
- Kondisi dengan beberapa kunci konteks atau nilai
- AWS kunci konteks kondisi global

Saran — Konfirmasikan jenis klaim audiens

Kode masalah: CONFIRM_AUDIENCE_CLAIM_TYPE

Jenis pencarian: SARAN

Detail temuan

Di dalam Konsol Manajemen AWS, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
Confirm audience claim type: The "{{key}}" ({{audienceType}}) claim key identifies
 the recipients that the JSON web token is intended for. Because this claim is single-
valued, do not use a qualifier.
```

Dalam panggilan terprogram ke AWS API AWS CLI or, temuan untuk pemeriksaan ini mencakup pesan berikut:

```
"findingDetails": "The "{{key}}" ({{audienceType}}) claim key identifies the recipients
 that the JSON web token is intended for. Because this claim is single-valued, do not
 use a qualifier."
```

Menyelesaikan saran

K aud unci klaim (audiens) adalah pengenal unik untuk aplikasi yang diberikan kepada Anda saat mendaftarkan aplikasi dengan IdP dan mengidentifikasi penerima yang menjadi tujuan token web JSON. Klaim audiens dapat bernilai banyak atau bernilai tunggal. Jika klaim bernilai banyak, gunakan operator set `ForAnyValue` kondisi `ForAllValues` atau. Jika klaim bernilai tunggal, jangan gunakan operator set kondisi.

Istilah terkait

- Membuat peran untuk identitas web atau OpenID Connect Federation (konsol)
- Kunci konteks multinilai
- Single-valued vs. kunci kondisi multinilai

Validasi kebijakan dengan pemeriksaan kebijakan khusus IAM Access Analyzer

Anda dapat menggunakan pemeriksaan kebijakan khusus untuk memeriksa akses baru berdasarkan standar keamanan Anda. Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

Memvalidasi kebijakan dengan pemeriksaan kebijakan khusus (konsol)

Sebagai langkah opsional, Anda dapat menjalankan pemeriksaan kebijakan khusus saat mengedit kebijakan di editor kebijakan JSON di konsol IAM. Anda dapat memeriksa apakah kebijakan yang diperbarui memberikan akses baru dibandingkan dengan versi yang ada.

Untuk memeriksa akses baru saat mengedit kebijakan IAM JSON

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan kebijakan yang ingin Anda edit. Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih tab Izin, lalu pilih Edit.
5. Pilih opsi JSON dan buat pembaruan kebijakan Anda.

6. Di panel validasi kebijakan di bawah kebijakan, pilih tab Periksa akses baru, lalu pilih Periksa kebijakan. Jika izin yang dimodifikasi memberikan akses baru, pernyataan akan disorot di panel validasi kebijakan.
7. Jika Anda tidak bermaksud memberikan akses baru, perbarui pernyataan kebijakan dan pilih Periksa kebijakan hingga tidak ada akses baru yang terdeteksi.

> **Note**
>
> Biaya dikaitkan dengan setiap cek untuk akses baru. Untuk detail selengkapnya tentang harga, lihat harga IAM Access Analyzer.

8. Pilih Berikutnya.
9. Pada halaman Tinjau dan simpan, tinjau Izin yang ditentukan dalam kebijakan ini, lalu pilih Simpan perubahan.

Memvalidasi kebijakan dengan pemeriksaan kebijakan khusus (AWS CLI atau API)

Anda dapat menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer dari AWS CLI atau IAM Access Analyzer API.

Untuk menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer ()AWS CLI

- Untuk memeriksa apakah akses baru diizinkan untuk kebijakan yang diperbarui jika dibandingkan dengan kebijakan yang ada, jalankan perintah berikut: check-no-new-access
- Untuk memeriksa apakah akses yang ditentukan tidak diizinkan oleh kebijakan, jalankan perintah berikut: check-access-not-granted
- Untuk memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu, jalankan perintah berikut: check-no-public-access

Untuk menjalankan pemeriksaan kebijakan kustom IAM Access Analyzer (API)

- Untuk memeriksa apakah akses baru diizinkan untuk kebijakan yang diperbarui jika dibandingkan dengan kebijakan yang ada, gunakan operasi CheckNoNewAccessAPI.
- Untuk memeriksa apakah akses yang ditentukan tidak diizinkan oleh kebijakan, gunakan operasi CheckAccessNotGrantedAPI.
- Untuk memeriksa apakah kebijakan sumber daya dapat memberikan akses publik ke jenis sumber daya tertentu, gunakan operasi CheckNoPublicAccessAPI.

Pembuatan kebijakan Penganalisis Akses IAM

Sebagai administrator atau pengembang, Anda dapat memberikan izin kepada entitas IAM (pengguna atau peran) di luar apa yang mereka butuhkan. IAM menyediakan beberapa opsi untuk membantu Anda memperbaiki izin yang Anda berikan.

Salah satu opsi adalah membuat kebijakan IAM yang didasarkan pada aktivitas akses untuk entitas. IAM Access Analyzer meninjau AWS CloudTrail log Anda dan menghasilkan templat kebijakan yang berisi izin yang digunakan entitas dalam rentang tanggal yang Anda tentukan. Anda dapat menggunakan templat untuk membuat kebijakan dengan izin mendetail yang memberikan hanya izin yang diperlukan untuk mendukung kasus penggunaan khusus Anda.

Topik

- Cara kerja pembuatan kebijakan
- Informasi tingkat tindakan dan layanan
- Hal yang perlu diketahui tentang pembuatan kebijakan
- Izin diperlukan untuk menghasilkan kebijakan
- Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (konsol)
- Buat kebijakan menggunakan AWS CloudTrail data di akun lain
- Buat kebijakan berdasarkan CloudTrail aktivitas (AWS CLI)
- Buat kebijakan berdasarkan CloudTrail aktivitas (AWS API)
- Layanan pembuatan kebijakan IAM Access Analyzer

Cara kerja pembuatan kebijakan

IAM Access Analyzer menganalisis CloudTrail peristiwa Anda untuk mengidentifikasi tindakan dan layanan yang telah digunakan oleh entitas IAM (pengguna atau peran). Kemudian menghasilkan kebijakan IAM yang didasarkan pada aktivitas itu. Anda dapat menyempurnakan izin entitas saat mengganti kebijakan izin luas yang dilampirkan ke entitas dengan kebijakan yang dibuat. Berikut ini adalah gambaran umum tingkat tinggi tentang proses pembuatan kebijakan.

- Menyiapkan pembuatan templat kebijakan — Anda menentukan periode waktu hingga 90 hari bagi IAM Access Analyzer untuk menganalisis AWS CloudTrail peristiwa historis Anda. Anda harus menentukan peran layanan yang sudah ada atau membuat peran layanan baru. Peran layanan memberi IAM Access Analyzer akses akses ke informasi CloudTrail jejak dan layanan yang terakhir diakses untuk mengidentifikasi layanan dan tindakan yang digunakan. Anda harus menentukan

CloudTrail jejak yang mencatat peristiwa untuk akun sebelum Anda dapat membuat kebijakan. Untuk informasi selengkapnya tentang kuota IAM Access Analyzer untuk CloudTrail data, lihat Kuota Penganalisis Ak ses IAM.

- Buat kebijakan — Penganalisis Akses IAM menghasilkan kebijakan berdasarkan aktivitas akses di CloudTrail acara Anda.
- Meninjau dan menyesuaikan kebijakan — Setelah kebijakan dibuat, Anda dapat meninjau layanan dan tindakan yang digunakan oleh entitas selama rentang tanggal yang ditentukan. Anda dapat lebih menyesuaikan kebijakan dengan menambahkan atau menghapus izin, menentukan sumber daya, dan menambahkan syarat ke templat kebijakan.
- Buat dan lampirkan kebijakan — Anda memiliki opsi untuk menyimpan kebijakan yang dihasilkan dengan membuat kebijakan terkelola. Anda dapat melampirkan kebijakan yang Anda buat untuk pengguna atau peran yang aktivitasnya digunakan untuk menghasilkan kebijakan.

Informasi tingkat tindakan dan layanan

Ketika IAM Access Analyzer membuat kebijakan IAM, informasi dikembalikan untuk membantu Anda menyesuaikan kebijakan lebih lanjut. Dua kategori informasi dapat dihasilkan ketika kebijakan yang membuat:

- Kebijakan dengan informasi tingkat tindakan — Untuk beberapa AWS layanan, seperti Amazon EC2, IAM Access Analyzer dapat mengidentifikasi tindakan yang ditemukan dalam CloudTrail acara Anda dan mencantumkan tindakan yang digunakan dalam kebijakan yang dihasilkannya. Untuk daftar layanan yang didukung, lihatLayanan pembuatan kebijakan IAM Access Analyzer. Untuk beberapa layanan, IAM Access Analyzer meminta Anda untuk menambahkan tindakan untuk layanan ke kebijakan yang dihasilkan.
- Kebijakan dengan informasi tingkat layanan — Penganalisis Akses IAM menggunakan informasi yang terakhir diakses untuk membuat template kebijakan dengan semua layanan yang baru-baru ini digunakan. Saat menggunakan Konsol Manajemen AWS, kami meminta Anda untuk meninjau layanan dan menambahkan tindakan untuk menyelesaikan kebijakan.

Untuk daftar tindakan di setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan di Referensi Otorisasi Layanan.

Hal yang perlu diketahui tentang pembuatan kebijakan

Sebelum Anda membuat kebijakan, tinjau detail penting berikut ini.

- Aktif CloudTrail kan jejak — Anda harus mengaktifkan CloudTrail jejak agar akun Anda dapat membuat kebijakan berdasarkan aktivitas akses. Saat Anda membuat CloudTrail jejak, CloudTrail mengirimkan peristiwa yang terkait dengan jejak Anda ke bucket Amazon S3 yang Anda tentukan. Untuk mempelajari cara membuat CloudTrail jejak, lihat Membuat jejak untuk AWS akun Anda di Panduan AWS CloudTrail Pengguna.
- Peristiwa data tidak tersedia — IAM Access Analyzer tidak mengidentifikasi aktivitas tingkat tindakan untuk peristiwa data, seperti peristiwa data Amazon S3, dalam kebijakan yang dihasilkan.
- PassRole— T `iam:PassRole` indakan tidak dilacak oleh CloudTrail dan tidak termasuk dalam kebijakan yang dihasilkan.
- Mengurangi waktu pembuatan kebijakan — Untuk membuat kebijakan lebih cepat, kurangi rentang tanggal yang Anda tentukan selama penyiapan untuk pembuatan kebijakan.
- Gunakan CloudTrail untuk audit — Jangan gunakan pembuatan kebijakan untuk tujuan audit; gunakan sebagai CloudTrail gantinya. Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihat Penc atatan panggilan IAM dan AWS STS API dengan AWS CloudTrail.
- Tindakan yang ditolak — Pembuatan kebijakan meninjau semua CloudTrail peristiwa, termasuk tindakan yang ditolak.
- Konsol IAM satu kebijakan — Anda dapat membuat satu kebijakan pada satu waktu di konsol IAM.
- Ketersediaan kebijakan yang dihasilkan Konsol IAM — Anda dapat meninjau kebijakan yang dihasilkan di konsol IAM hingga 7 hari setelah dibuat. Setelah 7 hari, Anda harus membuat kebijakan baru.
- Kuota pembuatan kebijakan — Untuk informasi tambahan tentang kuota pembuatan kebijakan Penganalisis Akses IAM, lihat Kuota Penganalisis Ak ses IAM.
- Tarif standar Amazon S3 berlaku — Saat Anda menggunakan fitur pembuatan kebijakan, IAM Access Analyzer meninjau CloudTrail log di bucket S3 Anda. Tidak ada biaya penyimpanan tambahan untuk mengakses CloudTrail log Anda untuk pembuatan kebijakan. AWS membebankan tarif standar Amazon S3 untuk permintaan dan transfer data CloudTrail log yang disimpan di bucket S3 Anda.
- AWS Control Tower dukungan — Pembuatan kebijakan tidak mendukung penggunaan AWS CloudTrail jejak yang AWS Control Tower dibuat saat membuat kebijakan, karena alasan berikut:
  - CloudTrail Data dari organisasi masuk ke akun lain, akun AWS Control Tower Log Archive.
  - Izin untuk bucket S3 tempat log ini disimpan tidak dapat dikonfigurasi ulang karena pembatasan pada bucket logging S3 yang ditetapkan oleh AWS Control Tower kebijakan kontrol layanan (SCP).

Izin diperlukan untuk menghasilkan kebijakan

Izin yang Anda butuhkan untuk membuat kebijakan untuk pertama kalinya berbeda dari yang Anda butuhkan untuk menghasilkan kebijakan untuk penggunaan berikutnya. Untuk informasi selengkapnya, lihat Memulai dengan AWS Identity and Access Management Access Analyzer.

First-time penyiapan

Ketika Anda membuat kebijakan untuk kali pertama, Anda harus memilih opsi Peran layanan yang sudah ada di akun Anda atau buat peran layanan baru. Peran layanan memberi IAM Access Analyzer akses CloudTrail dan layanan informasi yang terakhir diakses di akun Anda. Hanya administrator yang harus memiliki izin yang diperlukan untuk membuat dan mengonfigurasi peran. Oleh karena itu, kami merekomendasikan agar administrator menciptakan peran layanan selama penyiapan pertama kali. Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk membuat peran layanan, lihat Membuat peran untuk mendelegasikan izin ke AWS layanan.

Izin yang diperlukan untuk peran layanan

Bila Anda membuat peran layanan, Anda mengonfigurasi dua kebijakan untuk peran tersebut. Anda melampirkan Kebijakan izin IAM ke peran yang menentukan apa yang dapat dilakukan peran tersebut. Anda juga melampirkan kebijakan kepercayaan peran untuk peran yang menentukan penanggung jawab yang dapat menggunakan peran.

Contoh kebijakan pertama menunjukkan kebijakan izin untuk peran layanan yang diperlukan untuk menghasilkan kebijakan. Contoh kebijakan kedua menunjukkan kebijakan kepercayaan peran yang diperlukan untuk peran layanan. Anda dapat menggunakan kebijakan ini untuk membantu Anda membuat peran layanan saat menggunakan AWS API atau AWS CLI untuk membuat kebijakan. Saat Anda menggunakan konsol IAM untuk membuat peran layanan sebagai bagian dari proses pembuatan kebijakan, kami membuat kebijakan ini untuk Anda.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Kebijakan contoh berikut menunjukkan kebijakan kepercayaan peran dengan izin yang memungkinkan IAM Access Analyzer untuk mengambil peran tersebut.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Penggunaan selanjutnya

Untuk membuat kebijakan di Konsol Manajemen AWS, pengguna IAM harus memiliki kebijakan izin yang memungkinkan mereka untuk meneruskan peran layanan yang digunakan untuk pembuatan kebijakan ke Penganalisis Akses IAM. `iam:PassRole`biasanya disertai dengan `iam:GetRole` agar pengguna bisa mendapatkan rincian peran yang akan dilewati. Dalam contoh ini, pengguna hanya dapat meneruskan peran yang ada dalam akun yang ditentukan dengan nama yang dimulai dengan `AccessAnalyzerMonitorServiceRole*`. Untuk mempelajari selengkapnya tentang meneruskan peran IAM ke AWS layanan, lihat Memberikan izin pengguna untuk meneruskan peran ke AWS layanan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Anda juga harus memiliki izin Penganalisis Akses IAM berikut untuk membuat kebijakan di Konsol Manajemen AWS, AWS API, atau AWS CLI seperti yang ditunjukkan dalam pernyataan kebijakan berikut.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
```

```
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Untuk penggunaan pertama dan selanjutnya

Saat Anda menggunakan Konsol Manajemen AWS untuk membuat kebijakan, Anda harus memiliki `cloudtrail:ListTrails` izin untuk mencantumkan CloudTrail jejak di akun Anda seperti yang ditunjukkan dalam pernyataan kebijakan berikut.

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:ListTrails"
  ],
  "Resource": "*"
}
```

Menghasilkan kebijakan berdasarkan CloudTrail aktivitas (konsol)

Anda dapat membuat kebijakan untuk pengguna atau peran IAM.

Langkah 1: Buat kebijakan berdasarkan CloudTrail aktivitas

Prosedur berikut menjelaskan cara menghasilkan kebijakan untuk peran yang menggunakan Konsol Manajemen AWS.

Menghasilkan kebijakan untuk peran IAM

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi di sebelah kiri, pilih Peran.

> **Note**
>
> Langkah-langkah untuk membuat kebijakan berdasarkan aktivitas untuk pengguna IAM hampir identik. Untuk melakukannya, pilih Pengguna sebagai ganti Peran.

3. Dalam daftar peran di akun Anda, pilih nama peran yang aktivitasnya ingin Anda gunakan untuk menghasilkan kebijakan.
4. Pada tab Izin, di bagian Hasilkan kebijakan berdasarkan CloudTrail peristiwa, pilih Buat kebijakan.
5. Pada halaman Hasilkan kebijakan, tentukan periode waktu yang Anda inginkan agar IAM Access Analyzer menganalisis CloudTrail peristiwa Anda untuk tindakan yang diambil dengan peran tersebut. Anda dapat memilih rentang hingga 90 hari. Kami menyarankan Anda memilih periode waktu sesingkat mungkin untuk mengurangi waktu pembuatan kebijakan.
6. Di bagian CloudTrail akses, pilih peran yang sesuai atau buat peran baru jika peran yang sesuai tidak ada. Peran ini memberikan izin kepada IAM Access Analyzer untuk mengakses CloudTrail data atas nama Anda untuk meninjau aktivitas akses guna mengidentifikasi layanan dan tindakan yang telah digunakan. Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk peran ini, lihatIzin diperlukan untuk menghasilkan kebijakan.
7. Di bagian jejak CloudTrail yang akan dianalisis, tentukan jejak CloudTrail yang mencatat peristiwa untuk akun tersebut.

   Jika Anda memilih CloudTrail jejak yang menyimpan log di akun lain, kotak informasi tentang akses lintas akun akan ditampilkan. Cross-accountakses membutuhkan pengaturan tambahan. Untuk mempelajari lebih lanjut, lihat Choose a role for cross-account access nanti dalam topik ini.
8. Pilih Buat kebijakan.
9. Sementara pembuatan kebijakan sedang berlangsung, Anda akan dikembalikan ke halaman Ringkasan Peran tab Izin. Tunggu hingga status di bagian Detail permintaan kebijakanmenampilkan bagian Sukses, lalu pilih Lihat kebijakan yang dibuat. Anda dapat melihat kebijakan yang dihasilkan hingga tujuh hari. Jika Anda membuat kebijakan lain, kebijakan yang ada diganti dengan yang baru yang Anda buat.

Langkah 2: Tinjau izin dan tambahkan tindakan untuk layanan yang digunakan

Tinjau layanan dan tindakan yang diidentifikasi oleh IAM Access Analyzer sebagai peran yang digunakan. Anda dapat menambahkan tindakan untuk setiap layanan yang digunakan untuk templat kebijakan yang dihasilkan.

1. Tinjau bagian berikut:

- Pada Tinjau izin, tinjau daftarTindakan yang disertakan dalam kebijakan yang dihasilkan. Daftar ini menampilkan layanan dan tindakan yang diidentifikasi oleh IAM Access Analyzer digunakan oleh peran dalam rentang tanggal yang ditentukan.
- Bagian Layanan yang digunakan menampilkan layanan tambahan yang diidentifikasi oleh IAM Access Analyzer yang digunakan oleh peran dalam rentang tanggal yang ditentukan. Informasi tentang tindakan yang digunakan mungkin tidak tersedia untuk layanan yang tercantum dalam bagian ini. Gunakan menu untuk setiap layanan yang tercantum untuk secara manual memilih tindakan yang ingin Anda sertakan dalam kebijakan.

2. Setelah selesai menambahkan tindakan, pilih Next (Berikutnya).

Langkah 3: Selanjutnya sesuaikan kebijakan yang dihasilkan

Anda dapat lebih lanjut menyesuaikan kebijakan dengan menambahkan atau menghapus izin atau menentukan sumber daya.

Untuk menyesuaikan kebijakan yang dihasilkan

1. Memperbarui templat kebijakan. Template kebijakan berisi placeholder ARN sumber daya untuk tindakan yang mendukung izin tingkat sumber daya, seperti yang ditunjukkan pada gambar berikut. Resource-level izin mengacu pada kemampuan untuk menentukan sumber daya mana pengguna diizinkan untuk melakukan tindakan. Kami menyarankan agar Anda menggunakan ARN untuk menentukan sumber daya individu Anda dalam kebijakan untuk tindakan yang mendukung izin tingkat sumber daya. Anda dapat mengganti sumber daya pengganti ARN dengan ARN sumber daya yang valid untuk kasus penggunaan Anda.

   Jika suatu tindakan tidak mendukung izin tingkat sumber daya, Anda harus menggunakan wildcard (*) untuk menentukan bahwa semua sumber daya dapat dipengaruhi oleh tindakan tersebut. Untuk mempelajari AWS layanan mana yang mendukung izin tingkat sumber daya, lihat AWS layanan yang berfungsi dengan IAM. Untuk daftar tindakan di setiap layanan, dan untuk mempelajari tindakan mana yang mendukung izin tingkat sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Syarat untuk Layanan AWS.



2. (Opsional) Tambahkan, modifikasi, atau hapus pernyataan kebijakan JSON dalam templat. Untuk mempelajari lebih lanjut tentang menulis kebijakan JSON, lihat Membuat kebijakan IAM (konsol).
3. Setelah selesai menyesuaikan templat kebijakan, Anda akan memiliki opsi berikut:

   - (Opsional) Anda dapat menyalin JSON dalam templat untuk menggunakan secara terpisah di luar halaman Kebijakan yang dihasilkan. Misalnya, jika Anda ingin menggunakan JSON untuk membuat kebijakan di akun yang berbeda. Jika kebijakan dalam templat Anda melebihi batas 6.144 karakter untuk kebijakan JSON, kebijakan dibagi menjadi beberapa kebijakan.
   - Pilih Next (Berikutnya) untuk meninjau dan membuat kebijakan terkelola di akun yang sama.

Langkah 4: Tinjau dan buat kebijakan terkelola

Jika Anda memiliki izin untuk membuat dan melampirkan kebijakan IAM, Anda dapat membuat kebijakan terkelola dari kebijakan yang dibuat. Anda kemudian dapat melampirkan kebijakan ke pengguna atau peran di akun Anda.

Untuk meninjau dan membuat kebijakan

1. Pada halaman Tinjau dan buat kebijakan terkelola, masukkan Nama dan Deskripsi (opsional) untuk kebijakan yang Anda buat.

2. (Opsional) Di bagian Summary (Ringkasan), Anda dapat meninjau izin yang akan disertakan dalam kebijakan.
3. (Opsional) Tambahkan metadata ke kebijakan dengan melampirkan tanda sebagai pasangan nilai kunci. Untuk informasi lebih lanjut tentang penggunaan tanda dan IAM, lihat Penandaan sumber daya IAM.
4. Setelah selesai, lakukan salah satu hal berikut:
   - Anda dapat melampirkan kebijakan baru langsung ke peran yang digunakan untuk menghasilkan kebijakan. Untuk melakukan ini, di dekat bagian bawah halaman, pilih kotak centang di sebelah kebijakan Lamp irkan ke ***YourRoleName***. Kemudian pilih Create and attach policy (Membuat dan melampirkan kebijakan).
   - Atau, pilih Buat kebijakan. Anda dapat menemukan kebijakan yang Anda buat dalam daftar kebijakan di panel navigasi Kebijakan konsol IAM.
5. Anda dapat melampirkan kebijakan yang Anda buat ke entitas di akun. Setelah Anda melampirkan kebijakan, Anda dapat menghapus kebijakan lain yang terlalu luas yang mungkin dilampirkan ke entitas. Untuk mempelajari cara melampirkan kebijakan terkelola, lihat Men ambahkan izin identitas IAM (konsol).

Buat kebijakan menggunakan AWS CloudTrail data di akun lain

Anda dapat membuat CloudTrail jejak yang menyimpan data di akun pusat untuk merampingkan aktivitas pengelolaan. Misalnya, Anda dapat menggunakan AWS Organizations untuk membuat jejak yang mencatat semua peristiwa untuk semua Akun AWS dalam organisasi itu. Jejak itu milik akun pusat.

Jika Anda ingin membuat kebijakan untuk pengguna atau peran dalam akun yang berbeda dari akun tempat data CloudTrail log Anda disimpan, Anda harus memberikan akses lintas akun. Untuk melakukan ini, Anda memerlukan peran dan kebijakan bucket yang memberikan izin IAM Access Analyzer ke CloudTrail log Anda. Untuk informasi selengkapnya tentang membuat jalur Organisasi, lihat Membuat jejak untuk organisasi.

Dalam contoh ini, asumsikan bahwa Anda ingin membuat kebijakan untuk pengguna atau peran di akun A. CloudTrail Jejak di akun A menyimpan CloudTrail log dalam bucket di akun B. Sebelum Anda dapat membuat kebijakan, Anda harus membuat pembaruan berikut:

1. Pilih peran yang ada, atau buat peran layanan baru yang memberikan akses IAM Access Analyzer ke bucket di akun B (tempat CloudTrail log Anda disimpan).

2. Verifikasi kepemilikan objek bucket Amazon S3 dan kebijakan izin bucket di akun B sehingga IAM Access Analyzer dapat mengakses objek di bucket.

Langkah 1: Pilih atau buat peran untuk akses lintas akun

- Pada layar Hasilkan kebijakan, opsi untuk Gunakan peran yang ada dipilih sebelumnya untuk Anda jika peran dengan izin yang diperlukan ada di akun Anda. Jika tidak, pilih Buat dan gunakan peran layanan baru. Peran baru digunakan untuk memberikan akses IAM Access Analyzer ke CloudTrail log Anda di akun B.

Langkah 2: Verifikasi atau perbarui konfigurasi bucket Amazon S3 Anda di akun B

1. Masuk ke Konsol Manajemen AWS dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar Buckets, pilih nama bucket tempat log CloudTrail jejak Anda disimpan.
3. Pilih tab Izin dan buka bagian Kepemilikan Ob jek.

   Gunakan setelan bucket Kepemilikan Objek Amazon S3 untuk mengontrol kepemilikan objek yang Anda unggah ke bucket. Secara default, saat objek lain Akun AWS mengunggah ke bucket Anda, akun pengunggahan memiliki objek tersebut. Untuk membuat kebijakan, pemilik bucket harus memiliki semua objek dalam bucket. Tergantung pada kasus penggunaan ACL Anda, Anda mungkin perlu mengubah setelan Kepemilikan Ob jek untuk bucket Anda. Atur Kepemilikan Objek ke salah satu opsi berikut.

   - Pemilik bucket diberlakukan (disarankan)
   - Pemilik ember lebih disukai

   > **Important**
   >
   > Agar berhasil membuat kebijakan, objek dalam bucket harus dimiliki oleh pemilik bucket. Jika Anda memilih untuk menggunakan pilihan pemilik Bucket, Anda hanya dapat membuat kebijakan untuk jangka waktu setelah perubahan kepemilikan objek dilakukan.

Untuk mempelajari lebih lanjut tentang kepemilikan objek di Amazon S3, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda di Panduan Pengguna Amazon S3.

4. Tambahkan izin ke kebijakan bucket Amazon S3 Anda di akun B untuk mengizinkan akses peran di akun A.

   Kebijakan contoh berikut memungkinkan `ListBucket` dan `GetObject` untuk bucket bernama`amzn-s3-demo-bucket`. Ini memungkinkan akses jika peran yang mengakses bucket milik akun di organisasi Anda dan memiliki nama yang dimulai dengan`AccessAnalyzerMonitorServiceRole`. Menggunakan `aws:PrincipalArn` sebagai a `Condition` dalam `Resource` elemen memastikan bahwa peran hanya dapat mengakses aktivitas untuk akun jika itu milik akun A. Anda dapat mengganti `amzn-s3-demo-bucket` dengan nama bucket Anda, `optional-prefix` dengan awalan opsional untuk bucket, dan `organization-id` dengan ID organisasi Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "ArnLike": {
```

```
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Jika Anda mengenkripsi log menggunakan AWS KMS, perbarui kebijakan AWS KMS kunci Anda di akun tempat Anda menyimpan CloudTrail log untuk memberikan akses IAM Access Analyzer untuk menggunakan kunci Anda, seperti yang ditunjukkan pada contoh kebijakan berikut. Ganti `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` dengan ARN untuk jejak Anda dan `organization-id` dengan ID organisasi Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
            ]
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
     }
    ]
```

```
}
```

Buat kebijakan berdasarkan CloudTrail aktivitas (AWS CLI)

Anda dapat menggunakan perintah berikut untuk menghasilkan kebijakan menggunakan AWS CLI.

Untuk membuat kebijakan

- Pembuatan kebijakan awal aws accessanalyzer

Untuk melihat kebijakan yang dibuat

- kebijakan pembuatan akses aws accessanalyzer

Untuk membatalkan permintaan pembuatan kebijakan

- pembuatan kebijakan pembatalan aws accessanalyzer

Untuk melihat daftar permintaan pembuatan kebijakan

- generasi daftar kebijakan aws accessanalyzer

Buat kebijakan berdasarkan CloudTrail aktivitas (AWS API)

Anda dapat menggunakan operasi berikut untuk membuat kebijakan menggunakan AWS API.

Untuk membuat kebijakan

- StartPolicyGeneration

Untuk melihat kebijakan yang dibuat

- GetGeneratedPolicy

Untuk membatalkan permintaan pembuatan kebijakan

- CancelPolicyGeneration

Untuk melihat daftar permintaan pembuatan kebijakan

- ListPolicyGenerations

Layanan pembuatan kebijakan IAM Access Analyzer

Tabel berikut mencantumkan AWS layanan yang digunakan IAM Access Analyzer untuk membuat kebijakan dengan informasi tingkat tindakan. Untuk daftar tindakan di setiap layanan, lihat Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan di Referensi Otorisasi Layanan.

AWS menyediakan informasi pembuatan kebijakan dalam format JSON untuk merampingkan otomatisasi alur kerja manajemen kebijakan. Dengan informasi referensi layanan, Anda dapat mengakses informasi pembuatan kebijakan di seluruh file Layanan AWS yang dapat dibaca mesin. Untuk informasi selengkapnya, lihat Layanan AWS Informasi yang disederhanakan untuk akses terprogram di Referensi Otorisasi Layanan.

Layanan	Awalan layanan
AWS Identity and Access Management Access Analyzer	akses-analyzer
AWS Account Management	akun
AWS Certificate Manager	acm
Alur Kerja Terkelola Amazon untuk Apache Airflow	aliran udara
AWS Amplify	amplify
AWS Amplify Pembuat UI	amplifyuibuilder
Amazon AppIntegrations	integrasi aplikasi
AWS AppConfig	appconfig
Amazon AppFlow	Appflow
AWS Profiler Biaya Aplikasi	aplikasi-biaya-profiler
Wawasan CloudWatch Aplikasi Amazon	wawasan aplikasi

Layanan	Awalan layanan
AWS App Mesh	appmesh
WorkSpaces Aplikasi Amazon	appstream
AWS AppSync	appsync
Layanan Terkelola Amazon untuk Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	Pengelola Audit
AWS Auto Scaling	penskalaan otomatis
AWS Marketplace	aws-marketplace
AWS Backup	pencadangan
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	anggaran
AWS Cloud9	awan9
AWS CloudFormation	pembentukan awan
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch

Layanan	Awalan layanan
AWS CodeArtifact	kodeartefak
AWS CodeDeploy	penyebaran kode
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Peninjau Amazon	pengulas codeguru
AWS CodePipeline	codepipeline
AWS CodeStar	bintang kode
AWS CodeStar Pemberitahuan	pemberitahuan bintang kode
Identitas Amazon Cognito	kognito-identitas
Kumpulan pengguna Amazon Cognito	cognito-idp
Sinkronisasi Amazon Cognito	sinkronis asi kognito
Amazon Comprehend Medical	memahami-medis
AWS Compute Optimizer	pengoptim al komputasi
AWS Config	config
Pelanggan Amazon Connect	hubungkan
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	pertukaran data
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax

Layanan	Awalan layanan
AWS Device Farm	devicefarm
DevOpsGuru Amazon	devops-guru
AWS Direct Connect	DirectConnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Cluster Elastis Amazon DocumentDB	docdb-elastis
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Publik	ecr-publik
Layanan Kontainer Elastis Amazon	ecs
Layanan Amazon Elastic Kubernetes	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	tangkai beanstalk
Sistem File Elastis Amazon	sistem file elastis
Elastic Load Balancing	elasticlo adbalancing
Amazon EMR di EKS (Wadah EMR)	wadah emr
Amazon EMR Tanpa Server	emr-tanpa server
OpenSearch Layanan Amazon	es

Layanan	Awalan layanan
Amazon EventBridge	peristiwa
Amazon FinSpace	ruang finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	detektor penipuan
Amazon FSx	fsx
GameLift Peladen Amazon	gamelift
Amazon Location Service	geo
Amazon Glacier	gletser
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	stasiun tanah
Amazon GuardDuty	tugas jaga
AWS HealthLake	healthlake
Honeycode Amazon	kode madu
AWS Identity and Access Management	iam
AWS Toko Identitas	toko identitas
EC2 Image Builder	imagebuilder
Amazon Inspector Klasik	inspektur

Layanan	Awalan layanan
Amazon Inspector	inspektor2
AWS IoT	iot
AWS IoT Analytics	iotanalitik
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	pembuat iottwinmaker
AWS IoT Wireless	iotwireless
Layanan Video Interaktif Amazon	ivs
Obrolan Layanan Video Interaktif Amazon	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalitik
AWS Key Management Service	km
AWS Lambda	lambda
Amazon Lex V2	lex

Layanan	Awalan layanan
AWS License Manager Manajer Langganan Linux	lisensi-manajer-linux-langganan
Amazon Lightsail	lightsail
CloudWatch Log Amazon	log
Amazon Lookout for Equipment	peralatan pencarian
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Blockchain yang Dikelola Amazon	terkelolablockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Rekomendasi Strategi Migrasi Hub	Migration hub-strategi
Amazon Pinpoint	penargetan seluler
Amazon MQ	mq
AWS Network Manager	manajer jaringan
Studio Amazon Nimble	gesit

Layanan	Awalan layanan
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	pos terdepan
AWS Organizations	organisasi
AWS Wawasan Performa	pi
EventBridge Pipa Amazon	pipa
Amazon Polly	polly
Profil Pelanggan Pelanggan Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Tempat Sampah Daur Ulang	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API Data Pergeseran Merah Amazon	data pergeseran merah
AWS Migration Hub Refactor Spaces	ruang refaktor
Amazon Rekognition	rekognisi
AWS Resilience Hub	resiliencehub
Penjelajah Sumber Daya AWS	sumber daya-penjelajah-2

Layanan	Awalan layanan
AWS Resource Groups	kelompok sumber daya
AWS RoboMaker	robomaker
AWS Identity and Access Management Peran Di Mana Saja	peranandi mana saja
Rute Amazon 53	route53
Amazon Route 53 Kontrol Pemulihan	route53-recovery-control-config
Kesiapan Pemulihan Amazon Route 53	route53-p emulihan-kesiapan
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Layanan Penyimpanan Sederhana Amazon	s3
Amazon S3 on Outposts	pos terdepan s3
Kemampuan geospasial Amazon SageMaker AI	sagemaker-geospasial
Savings Plans	savingsplans
EventBridge Skema Amazon	skema
Amazon SimpleDB	sdb
AWS Secrets Manager	manajer rahasia
AWS Security Hub CSPM	securityhub
Amazon Security Lake	Securitylake

Layanan	Awalan layanan
AWS Serverless Application Repository	repo tanpa server
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Layanan Email Amazon Sederhana	ses
AWS Shield	melindungi
AWS Signer	penandatangan
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Amazon Pinpoint SMS dan Layanan Suara	sms-suara
AWS Snowball Edge	bola salju
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
Manajer Insiden AWS Systems Manager	insiden ssm
Manajer Sistem AWS untuk SAP	ssm-sap
AWS Step Functions	negara
AWS Security Token Service	sts
Layanan Alur Kerja Sederhana Amazon	swf
Amazon CloudWatch Synthetics	sintetis
AWS Resource Groups Tagging API	tanda

Layanan	Awalan layanan
Amazon Texttract	textract
Amazon Timestream	aliran waktu
AWS Pembangun Jaringan Telekomunikasi	tnb
Amazon Transcribe	mentranskripsikan
AWS Transfer Family	transfer
Amazon Translate	menerjemahkan
Connect ID Suara Pelanggan	voiceid
Kisi VPC Amazon	vpc-kisi
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Kebijaksanaan Pelanggan Amazon Connect	kebijaksanaan
Amazon WorkLink	tautan kerja
Amazon WorkSpaces	ruang kerja
AWS X-Ray	xray

Tindakan untuk informasi pembuatan kebijakan

Tabel berikut mencantumkan tindakan yang IAM Access Analyzer menghasilkan kebijakan dengan informasi tingkat tindakan.

> **Important**
>
> `iam:UpdateAccountName`Tindakan tersebut akan dihentikan pada 22 April 2026. Setelah 22 April 2026, hanya `account:PutAccountName` izin yang akan mengontrol akses pembaruan nama akun. Kami sangat menyarankan agar Anda memperbarui kebijakan

kontrol layanan (SCP) apa pun yang mengontrol pembaruan nama akun untuk menggunakan `account:PutAccountName` izin.

Awalan layanan	Tindakan
akses-analyzer	akses-analyzer: ApplyArchiveRule
	akses-analyzer: CancelPolicyGeneration
	akses-analyzer: CheckAccessNotGranted
	akses-analyzer: CheckNoNewAccess
	akses-analyzer: CheckNoPublicAccess
	akses-analyzer: CreateAccessPreview
	akses-analyzer: CreateAnalyzer
	akses-analyzer: CreateArchiveRule
	akses-analyzer: DeleteAnalyzer
	akses-analyzer: DeleteArchiveRule
	akses-analyzer: GenerateFindingRecommendation
	akses-analyzer: GetAccessPreview
	akses-analyzer: GetAnalyzedResource
	akses-analyzer: GetAnalyzer
	akses-analyzer: GetArchiveRule
	akses-analyzer: GetFinding
	akses-analyzer: GetFindingRecommendation
	akses-analyzer: GetFindingsStatistics

Awalan layanan	Tindakan
	akses-analyzer: GetGeneratedPolicy
	akses-analyzer: ListAccessPreviewFindings
	akses-analyzer: ListAccessPreviews
	akses-analyzer: ListAnalyzedResources
	akses-analyzer: ListAnalyzers
	akses-analyzer: ListArchiveRules
	akses-analyzer: ListFindings
	akses-analyzer: ListPolicyGenerations
	akses-analyzer: StartPolicyGeneration
	akses-analyzer: StartResourceScan
	akses-analyzer: UpdateAnalyzer
	akses-analyzer: UpdateArchiveRule
	akses-analyzer: UpdateFindings
	akses-analyzer: ValidatePolicy

Awalan layanan	Tindakan
akun	akun: AcceptPrimaryEmailUpdate akun: DeleteAlternateContact akun: DisableRegion akun: EnableRegion akun: GetAccountInformation akun: GetAlternateContact akun: GetContactInformation akun: GetGovCloudAccountInformation akun: GetPrimaryEmail akun: GetRegionOptStatus akun: ListRegions akun: PutAccountName akun: PutAlternateContact akun: PutContactInformation akun: StartPrimaryEmailUpdate

Awalan layanan	Tindakan
acm	ACM: CreateAcmeEndpoint
	ACM: DeleteCertificate
	ACM: DescribeCertificate
	ACM: ExportCertificate
	ACM: GetAccountConfiguration
	ACM: GetCertificate
	ACM: ImportCertificate
	ACM: ListAcmeEndpoints
	ACM: ListCertificates
	ACM: PutAccountConfiguration
	ACM: RenewCertificate
	ACM: RequestCertificate
	ACM: ResendValidationEmail
	ACM: SearchCertificates
	ACM: UpdateCertificateOptions

Awalan layanan	Tindakan
aliran udara	aliran udara: CreateCliToken aliran udara: CreateEnvironment aliran udara: CreateWebLoginToken aliran udara: DeleteEnvironment aliran udara: GetEnvironment aliran udara: ListEnvironments aliran udara: PublishMetrics aliran udara: UpdateEnvironment

Awalan layanan	Tindakan
amplify	memperkuat: CreateApp
	memperkuat: CreateBackendEnvironment
	memperkuat: CreateBranch
	memperkuat: CreateDeployment
	memperkuat: CreateDomainAssociation
	memperkuat: CreateWebHook
	memperkuat: DeleteApp
	memperkuat: DeleteBackendEnvironment
	memperkuat: DeleteBranch
	memperkuat: DeleteDomainAssociation
	memperkuat: DeleteJob
	memperkuat: DeleteWebHook
	memperkuat: GenerateAccessLogs
	memperkuat: GetApp
	memperkuat: GetArtifactUrl
	memperkuat: GetBackendEnvironment
	memperkuat: GetBranch
	memperkuat: GetDomainAssociation
	memperkuat: GetJob
	memperkuat: GetWebHook
	memperkuat: ListApps

Awalan layanan	Tindakan
	memperkuat: ListArtifacts
	memperkuat: ListBackendEnvironments
	memperkuat: ListBranches
	memperkuat: ListDomainAssociations
	memperkuat: ListJobs
	memperkuat: ListWebHooks
	memperkuat: StartDeployment
	memperkuat: StartJob
	memperkuat: StopJob
	memperkuat: UpdateApp
	memperkuat: UpdateBranch
	memperkuat: UpdateDomainAssociation
	memperkuat: UpdateWebHook

Awalan layanan	Tindakan
amplifyuibuilder	amplifyuibuilder: CreateComponent amplifyuibuilder: CreateForm amplifyuibuilder: CreateTheme amplifyuibuilder: DeleteComponent amplifyuibuilder: DeleteForm amplifyuibuilder: DeleteTheme amplifyuibuilder: ExportComponents amplifyuibuilder: ExportThemes amplifyuibuilder: GetCodegenJob amplifyuibuilder: ListCodegenJobs amplifyuibuilder: ListComponents amplifyuibuilder: ListForms amplifyuibuilder: ListThemes amplifyuibuilder: ResetMetadataFlag amplifyuibuilder: StartCodegenJob amplifyuibuilder: UpdateComponent amplifyuibuilder: UpdateForm amplifyuibuilder: UpdateTheme

Awalan layanan	Tindakan
integrasi aplikasi	integrasi aplikasi: CreateApplication
	integrasi aplikasi: CreateDataIntegration
	integrasi aplikasi: CreateDataIntegrationAssociation
	integrasi aplikasi: CreateEventIntegration
	integrasi aplikasi: DeleteApplication
	integrasi aplikasi: DeleteDataIntegration
	integrasi aplikasi: DeleteEventIntegration
	integrasi aplikasi: GetApplication
	integrasi aplikasi: GetDataIntegration
	integrasi aplikasi: GetEventIntegration
	integrasi aplikasi: ListApplicationAssociations
	integrasi aplikasi: ListApplications
	integrasi aplikasi: ListDataIntegrationAssociations
	integrasi aplikasi: ListDataIntegrations
	integrasi aplikasi: ListEventIntegrationAssociations
	integrasi aplikasi: ListEventIntegrations
	integrasi aplikasi: UpdateApplication
	integrasi aplikasi: UpdateDataIntegration
	integrasi aplikasi: UpdateDataIntegrationAssociation
	integrasi aplikasi: UpdateEventIntegration

Awalan layanan	Tindakan
appconfig	appconfig: CreateApplication appconfig: CreateConfigurationProfile appconfig: CreateDeploymentStrategy appconfig: CreateEnvironment appconfig: CreateExtension appconfig: CreateExtensionAssociation appconfig: CreateHostedConfigurationVersion appconfig: DeleteApplication appconfig: DeleteConfigurationProfile appconfig: DeleteDeploymentStrategy appconfig: DeleteEnvironment appconfig: DeleteExtension appconfig: DeleteExtensionAssociation appconfig: DeleteHostedConfigurationVersion appconfig: GetAccountSettings appconfig: GetApplication appconfig: GetConfiguration appconfig: GetConfigurationProfile appconfig: GetDeployment appconfig: GetDeploymentStrategy appconfig: GetEnvironment

Awalan layanan	Tindakan
	appconfig: GetExtension appconfig: GetExtensionAssociation appconfig: GetHostedConfigurationVersion appconfig: ListApplications appconfig: ListConfigurationProfiles appconfig: ListDeploymentStrategies appconfig: ListDeployments appconfig: ListEnvironments appconfig: ListExtensionAssociations appconfig: ListExtensions appconfig: ListHostedConfigurationVersions appconfig: StartDeployment appconfig: StopDeployment appconfig: UpdateAccountSettings appconfig: UpdateApplication appconfig: UpdateConfigurationProfile appconfig: UpdateDeploymentStrategy appconfig: UpdateEnvironment appconfig: UpdateExtension appconfig: UpdateExtensionAssociation appconfig: ValidateConfiguration

Awalan layanan	Tindakan
Appflow	Appflow: CancelFlowExecutions
	Appflow: CreateConnectorProfile
	Appflow: CreateFlow
	Appflow: DeleteConnectorProfile
	Appflow: DeleteFlow
	Appflow: DescribeConnector
	Appflow: DescribeConnectorEntity
	Appflow: DescribeConnectorProfiles
	Appflow: DescribeConnectors
	Appflow: DescribeFlow
	Appflow: DescribeFlowExecutionRecords
	Appflow: ListConnectorEntities
	Appflow: ListConnectors
	Appflow: ListFlows
	Appflow: RegisterConnector
	Appflow: ResetConnectorMetadataCache
	Appflow: StartFlow
	Appflow: StopFlow
	Appflow: UnRegisterConnector
	Appflow: UpdateConnectorProfile
	Appflow: UpdateConnectorRegistration

Awalan layanan	Tindakan
	Appflow: UpdateFlow

Awalan layanan	Tindakan
wawasan aplikasi	wawasan aplikasi: AddWorkload wawasan aplikasi: CreateApplication wawasan aplikasi: CreateComponent wawasan aplikasi: CreateLogPattern wawasan aplikasi: DeleteApplication wawasan aplikasi: DeleteComponent wawasan aplikasi: DeleteLogPattern wawasan aplikasi: DescribeApplication wawasan aplikasi: DescribeComponent wawasan aplikasi: DescribeComponentConfiguration wawasan aplikasi: DescribeComponentConfigurationRecommendation wawasan aplikasi: DescribeLogPattern wawasan aplikasi: DescribeObservation wawasan aplikasi: DescribeProblem wawasan aplikasi: DescribeProblemObservations wawasan aplikasi: DescribeWorkload wawasan aplikasi: ListApplications wawasan aplikasi: ListComponents wawasan aplikasi: ListConfigurationHistory wawasan aplikasi: ListLogPatternSets

Awalan layanan	Tindakan
	wawasan aplikasi: ListLogPatterns wawasan aplikasi: ListProblems wawasan aplikasi: ListWorkloads wawasan aplikasi: RemoveWorkload wawasan aplikasi: UpdateApplication wawasan aplikasi: UpdateComponent wawasan aplikasi: UpdateComponentConfiguration wawasan aplikasi: UpdateLogPattern wawasan aplikasi: UpdateWorkload

Awalan layanan	Tindakan
appmesh	appmesh: CreateGatewayRoute
	appmesh: CreateMesh
	appmesh: CreateRoute
	appmesh: CreateVirtualGateway
	appmesh: CreateVirtualNode
	appmesh: CreateVirtualRouter
	appmesh: CreateVirtualService
	appmesh: DeleteGatewayRoute
	appmesh: DeleteMesh
	appmesh: DeleteRoute
	appmesh: DeleteVirtualGateway
	appmesh: DeleteVirtualNode
	appmesh: DeleteVirtualRouter
	appmesh: DeleteVirtualService
	appmesh: DescribeGatewayRoute
	appmesh: DescribeMesh
	appmesh: DescribeRoute
	appmesh: DescribeVirtualGateway
	appmesh: DescribeVirtualNode
	appmesh: DescribeVirtualRouter
	appmesh: DescribeVirtualService

Awalan layanan	Tindakan
	appmesh: ListGatewayRoutes
	appmesh: ListMeshes
	appmesh: ListRoutes
	appmesh: ListVirtualGateways
	appmesh: ListVirtualNodes
	appmesh: ListVirtualRouters
	appmesh: ListVirtualServices
	appmesh: StreamAggregatedResources
	appmesh: UpdateGatewayRoute
	appmesh: UpdateMesh
	appmesh: UpdateRoute
	appmesh: UpdateVirtualGateway
	appmesh: UpdateVirtualNode
	appmesh: UpdateVirtualRouter
	appmesh: UpdateVirtualService

Awalan layanan	Tindakan
appstream	appstream: AssociateAppBlockBuilderAppBlock appstream: AssociateApplicationFleet appstream: AssociateApplicationToEntitlement appstream: AssociateFleet appstream: AssociateSoftwareToImageBuilder appstream: BatchAssociateUserStack appstream: BatchDisassociateUserStack appstream: CopyImage appstream: CreateAppBlock appstream: CreateAppBlockBuilder appstream: CreateAppBlockBuilderStreamingURL appstream: CreateApplication appstream: CreateDirectoryConfig appstream: CreateEntitlement appstream: CreateFleet appstream: CreateImageBuilder appstream: CreateImageBuilderStreamingURL appstream: CreateStack appstream: URL CreateStreaming appstream: CreateThemeForStack appstream: CreateUpdatedImage

Awalan layanan	Tindakan
	appstream: CreateUsageReportSubscription appstream: CreateUser appstream: DeleteAppBlock appstream: DeleteAppBlockBuilder appstream: DeleteApplication appstream: DeleteDirectoryConfig appstream: DeleteEntitlement appstream: DeleteFleet appstream: DeleteImage appstream: DeleteImageBuilder appstream: DeleteImagePermissions appstream: DeleteStack appstream: DeleteThemeForStack appstream: DeleteUsageReportSubscription appstream: DeleteUser appstream: DescribeAppBlockBuilderAppBlockAssociations appstream: DescribeAppBlockBuilders appstream: DescribeAppBlocks appstream: DescribeAppLicenseUsage appstream: DescribeApplicationFleetAssociations appstream: DescribeApplications

Awalan layanan	Tindakan
	appstream: DescribeDirectoryConfigs appstream: DescribeEntitlements appstream: DescribeFleets appstream: DescribeImageBuilders appstream: DescribeImagePermissions appstream: DescribeImages appstream: DescribeSessions appstream: DescribeStacks appstream: DescribeThemeForStack appstream: DescribeUsageReportSubscriptions appstream: DescribeUserStackAssociations appstream: DescribeUsers appstream: DisableUser appstream: DisassociateAppBlockBuilderAppBlock appstream: DisassociateApplicationFleet appstream: DisassociateApplicationFromEntitlement appstream: DisassociateFleet appstream: DisassociateSoftwareFromImageBuilder appstream: EnableUser appstream: ExpireSession appstream: GetExportImageTask

Awalan layanan	Tindakan
	appstream: ListAssociatedFleets appstream: ListAssociatedStacks appstream: ListEntitledApplications appstream: ListExportImageTasks appstream: StartAppBlockBuilder appstream: StartFleet appstream: StartImageBuilder appstream: StartSoftwareDeploymentToImageBuilder appstream: StopAppBlockBuilder appstream: StopFleet appstream: StopImageBuilder appstream: UpdateAppBlockBuilder appstream: UpdateApplication appstream: UpdateDirectoryConfig appstream: UpdateEntitlement appstream: UpdateFleet appstream: UpdateImagePermissions appstream: UpdateStack appstream: UpdateThemeForStack

Awalan layanan	Tindakan
appsync	appsync: AssociateApi
	appsync: AssociateMergedGraphqlApi
	appsync: AssociateSourceGraphqlApi
	appsync: CreateApi
	appsync: CreateApiCache
	appsync: CreateApiKey
	appsync: CreateChannelNamespace
	appsync: CreateDataSource
	appsync: CreateDomainName
	appsync: CreateFunction
	appsync: CreateGraphqlApi
	appsync: CreateResolver
	appsync: CreateType
	appsync: DeleteApi
	appsync: DeleteApiCache
	appsync: DeleteApiKey
	appsync: DeleteChannelNamespace
	appsync: DeleteDataSource
	appsync: DeleteDomainName
	appsync: DeleteFunction
	appsync: DeleteGraphqlApi

Awalan layanan	Tindakan
	appsync: DeleteResolver
	appsync: DeleteType
	appsync: DisassociateApi
	appsync: DisassociateMergedGraphqlApi
	appsync: DisassociateSourceGraphqlApi
	appsync: EvaluateCode
	appsync: EvaluateMappingTemplate
	appsync: FlushApiCache
	appsync: GetApi
	appsync: GetApiAssociation
	appsync: GetApiCache
	appsync: GetChannelNamespace
	appsync: GetDataSource
	appsync: GetDataSourceIntrospection
	appsync: GetDomainName
	appsync: GetFunction
	appsync: GetGraphqlApi
	appsync: GetGraphqlApiEnvironmentVariables
	appsync: GetIntrospectionSchema
	appsync: GetResolver
	appsync: GetSchemaCreationStatus

Awalan layanan	Tindakan
	appsync: GetSourceApiAssociation
	appsync: GetType
	appsync: ListApiKeys
	appsync: ListApis
	appsync: ListChannelNamespaces
	appsync: ListDataSources
	appsync: ListDomainNames
	appsync: ListFunctions
	appsync: ListGraphqlApis
	appsync: ListResolvers
	appsync: ListResolversByFunction
	appsync: ListSourceApiAssociations
	appsync: ListTypes
	appsync: ListTypesByAssociation
	appsync: PutGraphqlApiEnvironmentVariables
	appsync: StartDataSourceIntrospection
	appsync: StartSchemaCreation
	appsync: StartSchemaMerge
	appsync: UpdateApi
	appsync: UpdateApiCache
	appsync: UpdateApiKey

Awalan layanan	Tindakan
	appsync: UpdateChannelNamespace
	appsync: UpdateDataSource
	appsync: UpdateDomainName
	appsync: UpdateFunction
	appsync: UpdateGraphqlApi
	appsync: UpdateResolver
	appsync: UpdateSourceApiAssociation
	appsync: UpdateType

Awalan layanan	Tindakan
aps	aps: CreateAlertManagerDefinition aps: CreateAnomalyDetector aps: CreateLoggingConfiguration aps: CreateQueryLoggingConfiguration aps: CreateRuleGroupsNamespace aps: CreateWorkspace aps: DeleteAlertManagerDefinition aps: DeleteAnomalyDetector aps: DeleteLoggingConfiguration aps: DeleteQueryLoggingConfiguration aps: DeleteResourcePolicy aps: DeleteRuleGroupsNamespace aps: DeleteScraper aps: DeleteScraperLoggingConfiguration aps: DeleteWorkspace aps: DescribeAlertManagerDefinition aps: DescribeAnomalyDetector aps: DescribeLoggingConfiguration aps: DescribeQueryLoggingConfiguration aps: DescribeResourcePolicy aps: DescribeRuleGroupsNamespace

Awalan layanan	Tindakan
	aps: DescribeScraper aps: DescribeScraperLoggingConfiguration aps: DescribeWorkspace aps: DescribeWorkspaceConfiguration aps: GetDefaultScraperConfiguration aps: ListAnomalyDetectors aps: ListRuleGroupsNamespaces aps: ListScrapers aps: ListWorkspaces aps: PutAlertManagerDefinition aps: PutAnomalyDetector aps: PutResourcePolicy aps: PutRuleGroupsNamespace aps: UpdateLoggingConfiguration aps: UpdateQueryLoggingConfiguration aps: UpdateScraper aps: UpdateScraperLoggingConfiguration aps: UpdateWorkspaceAlias aps: UpdateWorkspaceConfiguration

Awalan layanan	Tindakan
athena	Athena: BatchGetNamedQuery
	Athena: BatchGetPreparedStatement
	Athena: BatchGetQueryExecution
	Athena: CancelCapacityReservation
	Athena: CreateCapacityReservation
	Athena: CreateDataCatalog
	Athena: CreateNamedQuery
	Athena: CreateNotebook
	Athena: CreatePreparedStatement
	Athena: CreatePresignedNotebookUrl
	Athena: CreateWorkGroup
	Athena: DeleteCapacityReservation
	Athena: DeleteDataCatalog
	Athena: DeleteNamedQuery
	Athena: DeleteNotebook
	Athena: DeletePreparedStatement
	Athena: DeleteWorkGroup
	Athena: ExportNotebook
	Athena: GetCalculationExecution
	Athena: GetCalculationExecutionCode
	Athena: GetCalculationExecutionStatus

Awalan layanan	Tindakan
	Athena: GetCapacityAssignmentConfiguration
	Athena: GetCapacityReservation
	Athena: GetDataCatalog
	Athena: GetDatabase
	Athena: GetNamedQuery
	Athena: GetNotebookMetadata
	Athena: GetPreparedStatement
	Athena: GetQueryExecution
	Athena: GetQueryResults
	Athena: GetQueryResultsStream
	Athena: GetQueryRuntimeStatistics
	Athena: GetResourceDashboard
	Athena: GetSession
	Athena: GetSessionEndpoint
	Athena: GetSessionStatus
	Athena: GetTableMetadata
	Athena: GetWorkGroup
	Athena: ImportNotebook
	Athena: ListApplicationDPUSizes
	Athena: ListCalculationExecutions
	Athena: ListCapacityReservations

Awalan layanan	Tindakan
	Athena: ListDataCatalogs
	Athena: ListDatabases
	Athena: ListEngineVersions
	Athena: ListExecutors
	Athena: ListNamedQueries
	Athena: ListNotebookMetadata
	Athena: ListNotebookSessions
	Athena: ListPreparedStatements
	Athena: ListQueryExecutions
	Athena: ListSessions
	Athena: ListTableMetadata
	Athena: ListWorkGroups
	Athena: PutCapacityAssignmentConfiguration
	Athena: StartCalculationExecution
	Athena: StartQueryExecution
	Athena: StartSession
	Athena: StopCalculationExecution
	Athena: StopQueryExecution
	Athena: TerminateSession
	Athena: UpdateCapacityReservation
	Athena: UpdateDataCatalog

Awalan layanan	Tindakan
	Athena: UpdateNamedQuery
	Athena: UpdateNotebook
	Athena: UpdateNotebookMetadata
	Athena: UpdatePreparedStatement
	Athena: UpdateWorkGroup

Awalan layanan	Tindakan
Pengelola Audit	Manajer Audit: AssociateAssessmentReportEvidenceFolder
	Manajer Audit: BatchAssociateAssessmentReportEvidence
	Manajer Audit: BatchCreateDelegationByAssessment
	Manajer Audit: BatchDeleteDelegationByAssessment
	Manajer Audit: BatchDisassociateAssessmentReportEvidence
	Manajer Audit: BatchImportEvidenceToAssessmentControl
	Manajer Audit: CreateAssessment
	Manajer Audit: CreateAssessmentFramework
	Manajer Audit: CreateAssessmentReport
	Manajer Audit: CreateControl
	Manajer Audit: DeleteAssessment
	Manajer Audit: DeleteAssessmentFramework
	Manajer Audit: DeleteAssessmentFrameworkShare
	Manajer Audit: DeleteAssessmentReport
	Manajer Audit: DeleteControl
	Manajer Audit: DeregisterAccount
	Manajer Audit: DeregisterOrganizationAdminAccount
	Manajer Audit: DisassociateAssessmentReportEvidenceFolder
	Manajer Audit: GetAccountStatus
	Manajer Audit: GetAssessment
	Manajer Audit: GetAssessmentFramework

Awalan layanan	Tindakan
	Manajer Audit: GetAssessmentReportUrl Manajer Audit: GetChangeLogs Manajer Audit: GetControl Manajer Audit: GetDelegations Manajer Audit: GetEvidence Manajer Audit: GetEvidenceByEvidenceFolder Manajer Audit: GetEvidenceFileUploadUrl Manajer Audit: GetEvidenceFolder Manajer Audit: GetEvidenceFoldersByAssessment Manajer Audit: GetEvidenceFoldersByAssessmentControl Manajer Audit: GetInsights Manajer Audit: GetInsightsByAssessment Manajer Audit: GetOrganizationAdminAccount Manajer Audit: GetServicesInScope Manajer Audit: GetSettings Manajer Audit: ListAssessmentControlInsightsByControlDomain Manajer Audit: ListAssessmentFrameworkShareRequests Manajer Audit: ListAssessmentFrameworks Manajer Audit: ListAssessmentReports Manajer Audit: ListAssessments Manajer Audit: ListControlDomainInsights

Awalan layanan	Tindakan
	Manajer Audit: ListControlDomainInsightsByAssessment Manajer Audit: ListControlInsightsByControlDomain Manajer Audit: ListControls Manajer Audit: ListKeywordsForDataSource Manajer Audit: ListNotifications Manajer Audit: RegisterAccount Manajer Audit: RegisterOrganizationAdminAccount Manajer Audit: StartAssessmentFrameworkShare Manajer Audit: UpdateAssessment Manajer Audit: UpdateAssessmentControl Manajer Audit: UpdateAssessmentControlSetStatus Manajer Audit: UpdateAssessmentFramework Manajer Audit: UpdateAssessmentFrameworkShare Manajer Audit: UpdateAssessmentStatus Manajer Audit: UpdateControl Manajer Audit: UpdateSettings Manajer Audit: ValidateAssessmentReportIntegrity

Awalan layanan	Tindakan
penskalaan otomatis	penskalaan otomatis: AttachInstances
	penskalaan otomatis: AttachLoadBalancerTargetGroups
	penskalaan otomatis: AttachLoadBalancers
	penskalaan otomatis: AttachTrafficSources
	penskalaan otomatis: BatchDeleteScheduledAction
	penskalaan otomatis: BatchPutScheduledUpdateGroupAction
	penskalaan otomatis: CancelInstanceRefresh
	penskalaan otomatis: CompleteLifecycleAction
	penskalaan otomatis: CreateAutoScalingGroup
	penskalaan otomatis: CreateLaunchConfiguration
	penskalaan otomatis: DeleteAutoScalingGroup
	penskalaan otomatis: DeleteLaunchConfiguration
	penskalaan otomatis: DeleteLifecycleHook
	penskalaan otomatis: DeleteNotificationConfiguration
	penskalaan otomatis: DeletePolicy
	penskalaan otomatis: DeleteScheduledAction
	penskalaan otomatis: DeleteWarmPool
	penskalaan otomatis: DescribeAccountLimits
	penskalaan otomatis: DescribeAdjustmentTypes
	penskalaan otomatis: DescribeAutoScalingGroups
	penskalaan otomatis: DescribeAutoScalingInstances

Awalan layanan	Tindakan
	penskalaan otomatis: DescribeAutoScalingNotificationTypes
	penskalaan otomatis: DescribeInstanceRefreshes
	penskalaan otomatis: DescribeLaunchConfigurations
	penskalaan otomatis: DescribeLifecycleHookTypes
	penskalaan otomatis: DescribeLifecycleHooks
	penskalaan otomatis: DescribeLoadBalancerTargetGroups
	penskalaan otomatis: DescribeLoadBalancers
	penskalaan otomatis: DescribeMetricCollectionTypes
	penskalaan otomatis: DescribeNotificationConfigurations
	penskalaan otomatis: DescribePolicies
	penskalaan otomatis: DescribeScalingActivities
	penskalaan otomatis: DescribeScalingProcessTypes
	penskalaan otomatis: DescribeScheduledActions
	penskalaan otomatis: DescribeTerminationPolicyTypes
	penskalaan otomatis: DescribeTrafficSources
	penskalaan otomatis: DescribeWarmPool
	penskalaan otomatis: DetachInstances
	penskalaan otomatis: DetachLoadBalancerTargetGroups
	penskalaan otomatis: DetachLoadBalancers
	penskalaan otomatis: DetachTrafficSources
	penskalaan otomatis: DisableMetricsCollection

Awalan layanan	Tindakan
	penskalaan otomatis: EnableMetricsCollection
	penskalaan otomatis: EnterStandby
	penskalaan otomatis: ExecutePolicy
	penskalaan otomatis: ExitStandby
	penskalaan otomatis: GetPredictiveScalingForecast
	penskalaan otomatis: PutLifecycleHook
	penskalaan otomatis: PutNotificationConfiguration
	penskalaan otomatis: PutScalingPolicy
	penskalaan otomatis: PutScheduledUpdateGroupAction
	penskalaan otomatis: PutWarmPool
	penskalaan otomatis: RecordLifecycleActionHeartbeat
	penskalaan otomatis: ResumeProcesses
	penskalaan otomatis: RollbackInstanceRefresh
	penskalaan otomatis: SetDesiredCapacity
	penskalaan otomatis: SetInstanceHealth
	penskalaan otomatis: SetInstanceProtection
	penskalaan otomatis: StartInstanceRefresh
	penskalaan otomatis: SuspendProcesses
	penskalaan otomatis: TerminateInstanceInAutoScalingGroup
	penskalaan otomatis: UpdateAutoScalingGroup
aws-marketplace	aws-pasar: GetEntitlements

Awalan layanan	Tindakan
pencadangan	cadangan: CancelLegalHold
	cadangan: CreateBackupPlan
	cadangan: CreateBackupSelection
	cadangan: CreateBackupVault
	cadangan: CreateFramework
	cadangan: CreateLegalHold
	cadangan: CreateLogicallyAirGappedBackupVault
	cadangan: CreateReportPlan
	cadangan: CreateRestoreAccessBackupVault
	cadangan: CreateRestoreTestingPlan
	cadangan: CreateRestoreTestingSelection
	cadangan: CreateTieringConfiguration
	cadangan: DeleteBackupPlan
	cadangan: DeleteBackupSelection
	cadangan: DeleteBackupVault
	cadangan: DeleteBackupVaultAccessPolicy
	cadangan: DeleteBackupVaultLockConfiguration
	cadangan: DeleteBackupVaultNotifications
	cadangan: DeleteFramework
	cadangan: DeleteRecoveryPoint
	cadangan: DeleteReportPlan

Awalan layanan	Tindakan
	cadangan: DeleteRestoreTestingPlan
	cadangan: DeleteRestoreTestingSelection
	cadangan: DeleteTieringConfiguration
	cadangan: DescribeBackupJob
	cadangan: DescribeBackupVault
	cadangan: DescribeCopyJob
	cadangan: DescribeFramework
	cadangan: DescribeGlobalSettings
	cadangan: DescribeProtectedResource
	cadangan: DescribeRecoveryPoint
	cadangan: DescribeRegionSettings
	cadangan: DescribeReportJob
	cadangan: DescribeReportPlan
	cadangan: DescribeRestoreJob
	cadangan: DescribeScanJob
	cadangan: DisassociateRecoveryPoint
	cadangan: DisassociateRecoveryPointFromParent
	cadangan: ExportBackupPlanTemplate
	cadangan: GetBackupPlan
	cadangan: GetBackupPlanFromJSON
	cadangan: GetBackupPlanFromTemplate

Awalan layanan	Tindakan
	cadangan: GetBackupSelection
	cadangan: GetBackupVaultAccessPolicy
	cadangan: GetBackupVaultNotifications
	cadangan: GetLegalHold
	cadangan: GetRecoveryPointRestoreMetadata
	cadangan: GetRestoreJobMetadata
	cadangan: GetRestoreTestingInferredMetadata
	cadangan: GetRestoreTestingPlan
	cadangan: GetRestoreTestingSelection
	cadangan: GetSupportedResourceTypes
	cadangan: GetTieringConfiguration
	cadangan: ListBackupJobSummaries
	cadangan: ListBackupJobs
	cadangan: ListBackupPlanTemplates
	cadangan: ListBackupPlanVersions
	cadangan: ListBackupPlans
	cadangan: ListBackupSelections
	cadangan: ListBackupVaults
	cadangan: ListCopyJobSummaries
	cadangan: ListCopyJobs
	cadangan: ListFrameworks

Awalan layanan	Tindakan
	cadangan: ListIndexedRecoveryPoints
	cadangan: ListLegalHolds
	cadangan: ListProtectedResources
	cadangan: ListRecoveryPointsByBackupVault
	cadangan: ListRecoveryPointsByLegalHold
	cadangan: ListRecoveryPointsByResource
	cadangan: ListReportJobs
	cadangan: ListReportPlans
	cadangan: ListRestoreAccessBackupVaults
	cadangan: ListRestoreJobSummaries
	cadangan: ListRestoreJobs
	cadangan: ListRestoreJobsByProtectedResource
	cadangan: ListRestoreTestingPlans
	cadangan: ListRestoreTestingSelections
	cadangan: ListScanJobSummaries
	cadangan: ListScanJobs
	cadangan: ListTieringConfigurations
	cadangan: PutBackupVaultAccessPolicy
	cadangan: PutBackupVaultLockConfiguration
	cadangan: PutBackupVaultNotifications
	cadangan: PutRestoreValidationResult

Awalan layanan	Tindakan
	cadangan: StartBackupJob
	cadangan: StartCopyJob
	cadangan: StartReportJob
	cadangan: StartRestoreJob
	cadangan: StopBackupJob
	cadangan: UpdateBackupPlan
	cadangan: UpdateFramework
	cadangan: UpdateGlobalSettings
	cadangan: UpdateRecoveryPointLifecycle
	cadangan: UpdateRegionSettings
	cadangan: UpdateReportPlan
	cadangan: UpdateRestoreTestingPlan
	cadangan: UpdateRestoreTestingSelection
	cadangan: UpdateTieringConfiguration

Awalan layanan	Tindakan
batch	batch: CancelJob batch: CreateComputeEnvironment batch: CreateConsumableResource batch: CreateJobQueue batch: CreateQuotaShare batch: CreateSchedulingPolicy batch: CreateServiceEnvironment batch: DeleteComputeEnvironment batch: DeleteConsumableResource batch: DeleteJobQueue batch: DeleteQuotaShare batch: DeleteSchedulingPolicy batch: DeleteServiceEnvironment batch: DeregisterJobDefinition batch: DescribeComputeEnvironments batch: DescribeConsumableResource batch: DescribeJobDefinitions batch: DescribeJobQueues batch: DescribeJobs batch: DescribeQuotaShare batch: DescribeSchedulingPolicies

Awalan layanan	Tindakan
	batch: DescribeServiceEnvironments batch: DescribeServiceJob batch: GetJobQueueSnapshot batch: ListConsumableResources batch: ListJobs batch: ListJobsByConsumableResource batch: ListQuotaShares batch: ListSchedulingPolicies batch: ListServiceJobs batch: RegisterJobDefinition batch: SubmitJob batch: SubmitServiceJob batch: TerminateJob batch: TerminateServiceJob batch: UpdateComputeEnvironment batch: UpdateConsumableResource batch: UpdateJobQueue batch: UpdateQuotaShare batch: UpdateSchedulingPolicy batch: UpdateServiceEnvironment batch: UpdateServiceJob

Awalan layanan	Tindakan
braket	Braket: CancelJob
	Braket: CancelQuantumTask
	Braket: CreateJob
	Braket: CreateQuantumTask
	Braket: CreateSpendingLimit
	Braket: GetDevice
	Braket: GetJob
	Braket: GetQuantumTask
	Braket: SearchDevices
	Braket: SearchJobs
	Braket: SearchQuantumTasks
	Braket: SearchSpendingLimits

Awalan layanan	Tindakan
anggaran	anggaran: CreateBudgetAction anggaran: DeleteBudgetAction anggaran: DescribeBudgetAction anggaran: DescribeBudgetActionHistories anggaran: DescribeBudgetActionsForAccount anggaran: DescribeBudgetActionsForBudget anggaran: ExecuteBudgetAction anggaran: ModifyBudget anggaran: UpdateBudgetAction anggaran: ViewBudget
awan9	cloud9: EC2 CreateEnvironment awan9: CreateEnvironmentMembership awan9: DeleteEnvironment awan9: DeleteEnvironmentMembership awan9: DescribeEnvironmentMemberships awan9: DescribeEnvironmentStatus awan9: DescribeEnvironments awan9: ListEnvironments awan9: UpdateEnvironment awan9: UpdateEnvironmentMembership

Awalan layanan	Tindakan
pembentukan awan	pembentukan awan: BatchDescribeTypeConfigurations pembentukan awan: CancelUpdateStack pembentukan awan: ContinueUpdateRollback pembentukan awan: CreateChangeSet pembentukan awan: CreateGeneratedTemplate pembentukan awan: CreateStack pembentukan awan: CreateStackInstances pembentukan awan: CreateStackSet pembentukan awan: DeactivateType pembentukan awan: DeleteChangeSet pembentukan awan: DeleteGeneratedTemplate pembentukan awan: DeleteStack pembentukan awan: DeleteStackInstances pembentukan awan: DeleteStackSet pembentukan awan: DeregisterType pembentukan awan: DescribeAccountLimits pembentukan awan: DescribeChangeSet pembentukan awan: DescribeChangeSetHooks pembentukan awan: DescribeEvents pembentukan awan: DescribeGeneratedTemplate pembentukan awan: DescribeOrganizationsAccess

Awalan layanan	Tindakan
	pembentukan awan: DescribePublisher pembentukan awan: DescribeResourceScan pembentukan awan: DescribeStackDriftDetectionStatus pembentukan awan: DescribeStackEvents pembentukan awan: DescribeStackInstance pembentukan awan: DescribeStackResource pembentukan awan: DescribeStackResourceDrifts pembentukan awan: DescribeStackResources pembentukan awan: DescribeStackSet pembentukan awan: DescribeStackSetOperation pembentukan awan: DescribeStacks pembentukan awan: DescribeType pembentukan awan: DescribeTypeRegistration pembentukan awan: DetectStackDrift pembentukan awan: DetectStackResourceDrift pembentukan awan: DetectStackSetDrift pembentukan awan: EstimateTemplateCost pembentukan awan: ExecuteChangeSet pembentukan awan: GetGeneratedTemplate pembentukan awan: GetHookResult pembentukan awan: GetStackPolicy

Awalan layanan	Tindakan
	pembentukan awan: GetTemplate
	pembentukan awan: GetTemplateSummary
	pembentukan awan: ImportStacksToStackSet
	pembentukan awan: ListChangeSets
	pembentukan awan: ListExports
	pembentukan awan: ListGeneratedTemplates
	pembentukan awan: ListHookResults
	pembentukan awan: ListImports
	pembentukan awan: ListResourceScanRelatedResources
	pembentukan awan: ListResourceScanResources
	pembentukan awan: ListResourceScans
	pembentukan awan: ListStackInstanceResourceDrifts
	pembentukan awan: ListStackInstances
	pembentukan awan: ListStackRefactors
	pembentukan awan: ListStackResources
	pembentukan awan: ListStackSetAutoDeploymentTargets
	pembentukan awan: ListStackSetOperationResults
	pembentukan awan: ListStackSetOperations
	pembentukan awan: ListStackSets
	pembentukan awan: ListTypeRegistrations
	pembentukan awan: ListTypeVersions

Awalan layanan	Tindakan
	pembentukan awan: ListTypes pembentukan awan: PublishType pembentukan awan: RecordHandlerProgress pembentukan awan: RegisterPublisher pembentukan awan: RegisterType pembentukan awan: RollbackStack pembentukan awan: SetStackPolicy pembentukan awan: SetTypeConfiguration pembentukan awan: SetTypeDefaultVersion pembentukan awan: SignalResource pembentukan awan: StartResourceScan pembentukan awan: StopStackSetOperation pembentukan awan: TestType pembentukan awan: UpdateGeneratedTemplate pembentukan awan: UpdateStack pembentukan awan: UpdateStackInstances pembentukan awan: UpdateStackSet pembentukan awan: UpdateTerminationProtection pembentukan awan: ValidateTemplate

Awalan layanan	Tindakan
cloudfront	cloudfront: AssociateAlias
	cloudfront: AssociateDistributionTenantWebACL
	cloudfront: AssociateDistributionWebACL
	cloudfront: CreateCachePolicy
	cloudfront: CreateCloudFrontOriginAccessIdentity
	cloudfront: CreateConnectionFunction
	cloudfront: CreateContinuousDeploymentPolicy
	cloudfront: CreateDistributionTenant
	cloudfront: CreateFieldLevelEncryptionConfig
	cloudfront: CreateFieldLevelEncryptionProfile
	cloudfront: CreateFunction
	cloudfront: CreateInvalidation
	cloudfront: CreateKeyGroup
	cloudfront: CreateKeyValueStore
	cloudfront: CreateMonitoringSubscription
	cloudfront: CreateOriginAccessControl
	cloudfront: CreateOriginRequestPolicy
	cloudfront: CreatePublicKey
	cloudfront: CreateRealtimeLogConfig
	cloudfront: CreateResponseHeadersPolicy
	cloudfront: CreateTrustStore

Awalan layanan	Tindakan
	cloudfront: DeleteAnycastIpList
	cloudfront: DeleteCachePolicy
	cloudfront: DeleteCloudFrontOriginAccessIdentity
	cloudfront: DeleteConnectionFunction
	cloudfront: DeleteConnectionGroup
	cloudfront: DeleteContinuousDeploymentPolicy
	cloudfront: DeleteDistribution
	cloudfront: DeleteDistributionTenant
	cloudfront: DeleteFieldLevelEncryptionConfig
	cloudfront: DeleteFieldLevelEncryptionProfile
	cloudfront: DeleteFunction
	cloudfront: DeleteKeyGroup
	cloudfront: DeleteKeyValueStore
	cloudfront: DeleteMonitoringSubscription
	cloudfront: DeleteOriginAccessControl
	cloudfront: DeleteOriginRequestPolicy
	cloudfront: DeletePublicKey
	cloudfront: DeleteRealtimeLogConfig
	cloudfront: DeleteResponseHeadersPolicy
	cloudfront: DeleteStreamingDistribution
	cloudfront: DeleteTrustStore

Awalan layanan	Tindakan
	cloudfront: DeleteVpcOrigin
	cloudfront: DescribeFunction
	cloudfront: DescribeKeyValueStore
	cloudfront: DisassociateDistributionTenantWebACL
	cloudfront: DisassociateDistributionWebACL
	cloudfront: GetAnycastIpList
	cloudfront: GetCachePolicy
	cloudfront: GetCachePolicyConfig
	cloudfront: GetCloudFrontOriginAccessIdentity
	cloudfront: GetCloudFrontOriginAccessIdentityConfig
	cloudfront: GetContinuousDeploymentPolicy
	cloudfront: GetContinuousDeploymentPolicyConfig
	cloudfront: GetDistributionConfig
	cloudfront: GetFieldLevelEncryption
	cloudfront: GetFieldLevelEncryptionConfig
	cloudfront: GetFieldLevelEncryptionProfile
	cloudfront: GetFieldLevelEncryptionProfileConfig
	cloudfront: GetFunction
	cloudfront: GetInvalidation
	cloudfront: GetInvalidationForDistributionTenant
	cloudfront: GetKeyGroup

Awalan layanan	Tindakan
	cloudfront: GetKeyGroupConfig
	cloudfront: GetMonitoringSubscription
	cloudfront: GetOriginAccessControl
	cloudfront: GetOriginAccessControlConfig
	cloudfront: GetOriginRequestPolicy
	cloudfront: GetOriginRequestPolicyConfig
	cloudfront: GetPublicKey
	cloudfront: GetPublicKeyConfig
	cloudfront: GetRealtimeLogConfig
	cloudfront: GetResponseHeadersPolicy
	cloudfront: GetResponseHeadersPolicyConfig
	cloudfront: GetStreamingDistribution
	cloudfront: GetStreamingDistributionConfig
	cloudfront: GetVpcOrigin
	cloudfront: ListAnycastIpLists
	cloudfront: ListCachePolicies
	cloudfront: ListCloudFrontOriginAccessIdentities
	cloudfront: ListConflictingAliases
	cloudfront: ListConnectionFunctions
	cloudfront: ListConnectionGroups
	cloudfront: ListContinuousDeploymentPolicies

Awalan layanan	Tindakan
	cloudfront: ListDistributionTenants
	cloudfront: ListDistributionTenantsByCustomization
	cloudfront: ListDistributions
	cloudfront: ListDistributionsByAnycastIpListId
	cloudfront: ListDistributionsByCachePolicyId
	cloudfront: ListDistributionsByConnectionMode
	cloudfront: ListDistributionsByKeyGroup
	cloudfront: ListDistributionsByOriginRequestPolicyId
	cloudfront: ListDistributionsByRealtimeLogConfig
	cloudfront: ListDistributionsByResponseHeadersPolicyId
	cloudfront: ListDistributionsByVpcOriginId
	cloudfront: ListDistributionsByWebACLId
	cloudfront: ListFieldLevelEncryptionConfigs
	cloudfront: ListFieldLevelEncryptionProfiles
	cloudfront: ListFunctions
	cloudfront: ListInvalidations
	cloudfront: ListInvalidationsForDistributionTenant
	cloudfront: ListKeyGroups
	cloudfront: ListKeyValueStores
	cloudfront: ListOriginAccessControls
	cloudfront: ListOriginRequestPolicies

Awalan layanan	Tindakan
	cloudfront: ListPublicKeys
	cloudfront: ListRealtimeLogConfigs
	cloudfront: ListResponseHeadersPolicies
	cloudfront: ListStreamingDistributions
	cloudfront: ListTrustStores
	cloudfront: PublishConnectionFunction
	cloudfront: PublishFunction
	cloudfront: TestConnectionFunction
	cloudfront: TestFunction
	cloudfront: UpdateAnycastIpList
	cloudfront: UpdateCachePolicy
	cloudfront: UpdateCloudFrontOriginAccessIdentity
	cloudfront: UpdateConnectionFunction
	cloudfront: UpdateConnectionGroup
	cloudfront: UpdateContinuousDeploymentPolicy
	cloudfront: UpdateDistribution
	cloudfront: UpdateDistributionTenant
	cloudfront: UpdateFieldLevelEncryptionConfig
	cloudfront: UpdateFieldLevelEncryptionProfile
	cloudfront: UpdateFunction
	cloudfront: UpdateKeyGroup

Awalan layanan	Tindakan
	cloudfront: UpdateKeyValueStore cloudfront: UpdateOriginAccessControl cloudfront: UpdateOriginRequestPolicy cloudfront: UpdatePublicKey cloudfront: UpdateRealtimeLogConfig cloudfront: UpdateResponseHeadersPolicy cloudfront: UpdateTrustStore
cloudhsm	cloudhsm: CreateHsm cloudhsm: DeleteBackup cloudhsm: DeleteHsm cloudhsm: DeleteResourcePolicy cloudhsm: DescribeBackups cloudhsm: DescribeClusters cloudhsm: GetResourcePolicy cloudhsm: InitializeCluster cloudhsm: ModifyBackupAttributes cloudhsm: ModifyCluster cloudhsm: PutResourcePolicy cloudhsm: RestoreBackup

Awalan layanan	Tindakan
cloudsearch	cloudsearch: BuildSuggesters
	cloudsearch: CreateDomain
	cloudsearch: DefineAnalysisScheme
	cloudsearch: DefineExpression
	cloudsearch: DefineIndexField
	cloudsearch: DefineSuggester
	cloudsearch: DeleteAnalysisScheme
	cloudsearch: DeleteDomain
	cloudsearch: DeleteExpression
	cloudsearch: DeleteIndexField
	cloudsearch: DeleteSuggester
	cloudsearch: DescribeAnalysisSchemes
	cloudsearch: DescribeAvailabilityOptions
	cloudsearch: DescribeDomainEndpointOptions
	cloudsearch: DescribeDomains
	cloudsearch: DescribeExpressions
	cloudsearch: DescribeIndexFields
	cloudsearch: DescribeScalingParameters
	cloudsearch: DescribeServiceAccessPolicies
	cloudsearch: DescribeSuggesters
	cloudsearch: IndexDocuments

Awalan layanan	Tindakan
	cloudsearch: ListDomainNames
	cloudsearch: UpdateAvailabilityOptions
	cloudsearch: UpdateDomainEndpointOptions
	cloudsearch: UpdateScalingParameters
	cloudsearch: UpdateServiceAccessPolicies

Awalan layanan	Tindakan
cloudtrail	cloudtrail: CancelQuery cloudtrail: CreateChannel cloudtrail: CreateDashboard cloudtrail: CreateEventDataStore cloudtrail: CreateTrail cloudtrail: DeleteChannel cloudtrail: DeleteDashboard cloudtrail: DeleteEventDataStore cloudtrail: DeleteResourcePolicy cloudtrail: DeleteTrail cloudtrail: DeregisterOrganizationDelegatedAdmin cloudtrail: DescribeQuery cloudtrail: DescribeTrails cloudtrail: DisableFederation cloudtrail: GenerateQuery cloudtrail: GetChannel cloudtrail: GetDashboard cloudtrail: GetEventConfiguration cloudtrail: GetEventDataStore cloudtrail: GetEventDataStoreData cloudtrail: GetEventSelectors

Awalan layanan	Tindakan
	cloudtrail: GetImport cloudtrail: GetInsightSelectors cloudtrail: GetResourcePolicy cloudtrail: GetTrail cloudtrail: GetTrailStatus cloudtrail: ListChannels cloudtrail: ListDashboards cloudtrail: ListEventDataStores cloudtrail: ListImportFailures cloudtrail: ListImports cloudtrail: ListInsightsData cloudtrail: ListPublicKeys cloudtrail: ListQueries cloudtrail: ListTrails cloudtrail: LookupEvents cloudtrail: PutEventConfiguration cloudtrail: PutEventSelectors cloudtrail: PutInsightSelectors cloudtrail: PutResourcePolicy cloudtrail: RegisterOrganizationDelegatedAdmin cloudtrail: RestoreEventDataStore

Awalan layanan	Tindakan
	cloudtrail: SearchSampleQueries cloudtrail: StartEventDataStoreIngestion cloudtrail: StartImport cloudtrail: StartLogging cloudtrail: StartQuery cloudtrail: StopEventDataStoreIngestion cloudtrail: StopImport cloudtrail: StopLogging cloudtrail: UpdateChannel cloudtrail: UpdateDashboard cloudtrail: UpdateEventDataStore cloudtrail: UpdateTrail

Awalan layanan	Tindakan
cloudwatch	jam tangan awan: DeleteAlarmMuteRule
	jam tangan awan: DeleteAlarms
	jam tangan awan: DeleteAnomalyDetector
	jam tangan awan: DeleteDashboards
	jam tangan awan: DeleteInsightRules
	jam tangan awan: DeleteMetricStream
	jam tangan awan: DescribeAlarmHistory
	jam tangan awan: DescribeAlarms
	jam tangan awan: DescribeAlarmsForMetric
	jam tangan awan: DescribeAnomalyDetectors
	jam tangan awan: DescribeInsightRules
	jam tangan awan: DisableAlarmActions
	jam tangan awan: DisableInsightRules
	jam tangan awan: EnableAlarmActions
	jam tangan awan: EnableInsightRules
	jam tangan awan: GetAlarmMuteRule
	jam tangan awan: GetDashboard
	jam tangan awan: GetDataset
	jam tangan awan: GetInsightRuleReport
	jam tangan awan: GetMetricStatistics
	jam tangan awan: GetMetricStream

Awalan layanan	Tindakan
	jam tangan awan: ListAlarmMuteRules
	jam tangan awan: ListDashboards
	jam tangan awan: ListManagedInsightRules
	jam tangan awan: ListMetricStreams
	jam tangan awan: PutAlarmMuteRule
	jam tangan awan: PutAnomalyDetector
	jam tangan awan: PutCompositeAlarm
	jam tangan awan: PutDashboard
	jam tangan awan: PutInsightRule
	jam tangan awan: PutLogAlarm
	jam tangan awan: PutManagedInsightRules
	jam tangan awan: PutMetricAlarm
	jam tangan awan: PutMetricStream
	jam tangan awan: SetAlarmState
	jam tangan awan: StartMetricStreams
	jam tangan awan: StopMetricStreams

Awalan layanan	Tindakan
kodeartefak	codeartefak: AssociateExternalConnection codeartefak: CopyPackageVersions codeartefak: CreateDomain codeartefak: CreateRepository codeartefak: DeleteDomain codeartefak: DeleteDomainPermissionsPolicy codeartefak: DeletePackage codeartefak: DeletePackageVersions codeartefak: DeleteRepository codeartefak: DeleteRepositoryPermissionsPolicy codeartefak: DescribeDomain codeartefak: DescribePackage codeartefak: DescribePackageVersion codeartefak: DescribeRepository codeartefak: DisassociateExternalConnection codeartefak: DisposePackageVersions codeartefak: GetAssociatedPackageGroup codeartefak: GetAuthorizationToken codeartefak: GetDomainPermissionsPolicy codeartefak: GetPackageVersionAsset codeartefak: GetPackageVersionReadme

Awalan layanan	Tindakan
	codeartefak: GetRepositoryEndpoint codeartefak: GetRepositoryPermissionsPolicy codeartefak: ListDomains codeartefak: ListPackageGroups codeartefak: ListPackageVersionAssets codeartefak: ListPackageVersionDependencies codeartefak: ListPackageVersions codeartefak: ListPackages codeartefak: ListRepositories codeartefak: ListRepositoriesInDomain codeartefak: PublishPackageVersion codeartefak: PutDomainPermissionsPolicy codeartefak: PutPackageMetadata codeartefak: PutPackageOriginConfiguration codeartefak: PutRepositoryPermissionsPolicy codeartefak: ReadFromRepository codeartefak: UpdatePackageVersionsStatus codeartefak: UpdateRepository

Awalan layanan	Tindakan
penyebaran kode	penyebaran kode: BatchGetApplicationRevisions
	penyebaran kode: BatchGetApplications
	penyebaran kode: BatchGetDeploymentGroups
	penyebaran kode: BatchGetDeploymentInstances
	penyebaran kode: BatchGetDeploymentTargets
	penyebaran kode: BatchGetDeployments
	penyebaran kode: BatchGetOnPremisesInstances
	penyebaran kode: ContinueDeployment
	penyebaran kode: CreateApplication
	penyebaran kode: CreateDeployment
	penyebaran kode: CreateDeploymentConfig
	penyebaran kode: CreateDeploymentGroup
	penyebaran kode: DeleteApplication
	penyebaran kode: DeleteDeploymentConfig
	penyebaran kode: DeleteDeploymentGroup
	penyebaran kode: DeleteGitHubAccountToken
	penyebaran kode: DeleteResourcesByExternalId
	penyebaran kode: DeregisterOnPremisesInstance
	penyebaran kode: GetApplication
	penyebaran kode: GetApplicationRevision
	penyebaran kode: GetDeployment

Awalan layanan	Tindakan
	penyebaran kode: GetDeploymentConfig
	penyebaran kode: GetDeploymentGroup
	penyebaran kode: GetDeploymentInstance
	penyebaran kode: GetDeploymentTarget
	penyebaran kode: GetOnPremisesInstance
	penyebaran kode: ListApplicationRevisions
	penyebaran kode: ListApplications
	penyebaran kode: ListDeploymentConfigs
	penyebaran kode: ListDeploymentGroups
	penyebaran kode: ListDeploymentInstances
	penyebaran kode: ListDeploymentTargets
	penyebaran kode: ListDeployments
	penyebaran kode: ListGitHubAccountTokenNames
	penyebaran kode: ListOnPremisesInstances
	penyebaran kode: PutLifecycleEventHookExecutionStatus
	penyebaran kode: RegisterApplicationRevision
	penyebaran kode: RegisterOnPremisesInstance
	penyebaran kode: SkipWaitTimeForInstanceTermination
	penyebaran kode: StopDeployment
	penyebaran kode: UpdateApplication
	penyebaran kode: UpdateDeploymentGroup

Awalan layanan	Tindakan
codeguru-profiler	codeguru-profiler: AddNotificationChannels codeguru-profiler: BatchGetFrameMetricData codeguru-profiler: CreateProfilingGroup codeguru-profiler: DeleteProfilingGroup codeguru-profiler: DescribeProfilingGroup codeguru-profiler: GetFindingsReportAccountSummary codeguru-profiler: GetNotificationConfiguration codeguru-profiler: GetPolicy codeguru-profiler: GetProfile codeguru-profiler: GetRecommendations codeguru-profiler: ListFindingsReports codeguru-profiler: ListProfileTimes codeguru-profiler: ListProfilingGroups codeguru-profiler: PutPermission codeguru-profiler: RemoveNotificationChannel codeguru-profiler: RemovePermission codeguru-profiler: SubmitFeedback codeguru-profiler: UpdateProfilingGroup

Awalan layanan	Tindakan
pengulas codeguru	pengulas codeguru: AssociateRepository
	pengulas codeguru: CreateCodeReview
	pengulas codeguru: DescribeCodeReview
	pengulas codeguru: DescribeRecommendationFeedback
	pengulas codeguru: DescribeRepositoryAssociation
	pengulas codeguru: DisassociateRepository
	pengulas codeguru: ListCodeReviews
	pengulas codeguru: ListRecommendationFeedback
	pengulas codeguru: ListRecommendations
	pengulas codeguru: ListRepositoryAssociations
	pengulas codeguru: PutRecommendationFeedback

Awalan layanan	Tindakan
codepipeline	codepipeline: AcknowledgeJob codepipeline: AcknowledgeThirdPartyJob codepipeline: CreateCustomActionType codepipeline: CreatePipeline codepipeline: DeleteCustomActionType codepipeline: DeletePipeline codepipeline: DeleteWebhook codepipeline: DeregisterWebhookWithThirdParty codepipeline: GetActionType codepipeline: GetJobDetails codepipeline: GetPipeline codepipeline: GetPipelineExecution codepipeline: GetPipelineState codepipeline: GetThirdPartyJobDetails codepipeline: ListActionExecutions codepipeline: ListActionTypes codepipeline: ListPipelineExecutions codepipeline: ListPipelines codepipeline: ListRuleExecutions codepipeline: ListRuleTypes codepipeline: ListWebhooks

Awalan layanan	Tindakan
	codepipeline: OverrideStageCondition codepipeline: PollForJobs codepipeline: PollForThirdPartyJobs codepipeline: PutActionRevision codepipeline: PutApprovalResult codepipeline: PutJobFailureResult codepipeline: PutJobSuccessResult codepipeline: PutThirdPartyJobFailureResult codepipeline: PutThirdPartyJobSuccessResult codepipeline: PutWebhook codepipeline: RegisterWebhookWithThirdParty codepipeline: RollbackStage codepipeline: StartPipelineExecution codepipeline: StopPipelineExecution codepipeline: UpdateActionType codepipeline: UpdatePipeline

Awalan layanan	Tindakan
bintang kode	bintang kode: AssociateTeamMember bintang kode: CreateProject bintang kode: CreateUserProfile bintang kode: DeleteProject bintang kode: DeleteUserProfile bintang kode: DescribeProject bintang kode: DescribeUserProfile bintang kode: DisassociateTeamMember bintang kode: ListProjects bintang kode: ListResources bintang kode: ListTeamMembers bintang kode: ListUserProfiles bintang kode: UpdateProject bintang kode: UpdateTeamMember bintang kode: UpdateUserProfile

Awalan layanan	Tindakan
pemberitahuan bintang kode	pemberitahuan bintang kode: CreateNotificationRule pemberitahuan bintang kode: DeleteNotificationRule pemberitahuan bintang kode: DeleteTarget pemberitahuan bintang kode: DescribeNotificationRule pemberitahuan bintang kode: ListEventTypes pemberitahuan bintang kode: ListNotificationRules pemberitahuan bintang kode: ListTargets Codestar-notifikasi:Berlangganan Codestar-notifications:berhenti berlangganan pemberitahuan bintang kode: UpdateNotificationRule

Awalan layanan	Tindakan
kognito-identitas	kognito-identitas: CreateIdentityPool
	kognito-identitas: DeleteIdentities
	kognito-identitas: DeleteIdentityPool
	kognito-identitas: DescribeIdentity
	kognito-identitas: DescribeIdentityPool
	kognito-identitas: GetIdentityPoolRoles
	kognito-identitas: ListIdentities
	kognito-identitas: ListIdentityPools
	kognito-identitas: LookupDeveloperIdentity
	kognito-identitas: MergeDeveloperIdentities
	kognito-identitas: SetIdentityPoolRoles
	kognito-identitas: UnlinkDeveloperIdentity
	kognito-identitas: UpdateIdentityPool

Awalan layanan	Tindakan
cognito-idp	cognito-idp: AddCustomAttributes cognito-idp: AddUserPoolClientSecret cognito-idp: AdminAddUserToGroup cognito-idp: AdminConfirmSignUp cognito-idp: AdminCreateUser cognito-idp: AdminDeleteUser cognito-idp: AdminDeleteUserAttributes cognito-idp: AdminDisableProviderForUser cognito-idp: AdminDisableUser cognito-idp: AdminEnableUser cognito-idp: AdminForgetDevice cognito-idp: AdminGetDevice cognito-idp: AdminGetUser cognito-idp: AdminGetUserAuthFactors cognito-idp: AdminInitiateAuth cognito-idp: AdminLinkProviderForUser cognito-idp: AdminListDevices cognito-idp: AdminListGroupsForUser cognito-idp: AdminListUserAuthEvents cognito-idp: AdminRemoveUserFromGroup cognito-idp: AdminResetUserPassword

Awalan layanan	Tindakan
	cognito-idp: AdminRespondToAuthChallenge cognito-idp: AdminSetUserMFAPreference cognito-idp: AdminSetUserPassword cognito-idp: AdminSetUserSettings cognito-idp: AdminUpdateAuthEventFeedback cognito-idp: AdminUpdateDeviceStatus cognito-idp: AdminUpdateUserAttributes cognito-idp: AdminUserGlobalSignOut cognito-idp: AssociateSoftwareToken cognito-idp: ChangePassword cognito-idp: ConfirmDevice cognito-idp: ConfirmForgotPassword cognito-idp: ConfirmSignUp cognito-idp: CreateGroup cognito-idp: CreateIdentityProvider cognito-idp: CreateManagedLoginBranding cognito-idp: CreateResourceServer cognito-idp: CreateTerms cognito-idp: CreateUserImportJob cognito-idp: CreateUserPool cognito-idp: CreateUserPoolClient

Awalan layanan	Tindakan
	cognito-idp: CreateUserPoolDomain cognito-idp: CreateUserPoolReplica cognito-idp: DeleteGroup cognito-idp: DeleteIdentityProvider cognito-idp: DeleteManagedLoginBranding cognito-idp: DeleteResourceServer cognito-idp: DeleteTerms cognito-idp: DeleteUser cognito-idp: DeleteUserAttributes cognito-idp: DeleteUserPool cognito-idp: DeleteUserPoolClient cognito-idp: DeleteUserPoolClientSecret cognito-idp: DeleteUserPoolDomain cognito-idp: DeleteUserPoolReplica cognito-idp: DescribeIdentityProvider cognito-idp: DescribeManagedLoginBranding cognito-idp: DescribeManagedLoginBrandingByClient cognito-idp: DescribeResourceServer cognito-idp: DescribeRiskConfiguration cognito-idp: DescribeTerms cognito-idp: DescribeUserImportJob

Awalan layanan	Tindakan
	cognito-idp: DescribeUserPool cognito-idp: DescribeUserPoolClient cognito-idp: DescribeUserPoolDomain cognito-idp: ForgetDevice cognito-idp: ForgotPassword Cognito-IDP:GetCSVHeader cognito-idp: GetDevice cognito-idp: GetGroup cognito-idp: GetIdentityProviderByIdentifier cognito-idp: GetLogDeliveryConfiguration cognito-idp: GetProvisionedLimit cognito-idp: GetSigningCertificate Cognito-IDP:getuicustomisasi cognito-idp: GetUser cognito-idp: GetUserAttributeVerificationCode cognito-idp: GetUserPoolMfaConfig cognito-idp: GlobalSignOut cognito-idp: InitiateAuth cognito-idp: ListDevices cognito-idp: ListGroups cognito-idp: ListIdentityProviders

Awalan layanan	Tindakan
	cognito-idp: ListResourceServers cognito-idp: ListTerms cognito-idp: ListUserImportJobs cognito-idp: ListUserPoolClientSecrets cognito-idp: ListUserPoolClients cognito-idp: ListUserPoolReplicas cognito-idp: ListUserPools cognito-idp: ListUsers cognito-idp: ListUsersInGroup cognito-idp: ResendConfirmationCode cognito-idp: RespondToAuthChallenge cognito-idp: RevokeToken cognito-idp: SetLogDeliveryConfiguration cognito-idp: SetRiskConfiguration Cognito-IDP: Setuicustomisasi cognito-idp: SetUserMFAPreference cognito-idp: SetUserPoolMfaConfig cognito-idp: SetUserSettings cognito-idp: SignUp cognito-idp: StartUserImportJob cognito-idp: StopUserImportJob

Awalan layanan	Tindakan
	cognito-idp: UpdateAuthEventFeedback cognito-idp: UpdateDeviceStatus cognito-idp: UpdateGroup cognito-idp: UpdateIdentityProvider cognito-idp: UpdateProvisionedLimit cognito-idp: UpdateResourceServer cognito-idp: UpdateTerms cognito-idp: UpdateUserAttributes cognito-idp: UpdateUserPool cognito-idp: UpdateUserPoolClient cognito-idp: UpdateUserPoolDomain cognito-idp: UpdateUserPoolReplica cognito-idp: VerifySoftwareToken cognito-idp: VerifyUserAttribute

Awalan layanan	Tindakan
sinkronisasi kognito	sinkronisasi kognito: BulkPublish
	sinkronisasi kognito: DeleteDataset
	sinkronisasi kognito: DescribeDataset
	sinkronisasi kognito: DescribeIdentityPoolUsage
	sinkronisasi kognito: DescribeIdentityUsage
	sinkronisasi kognito: GetBulkPublishDetails
	sinkronisasi kognito: GetCognitoEvents
	sinkronisasi kognito: GetIdentityPoolConfiguration
	sinkronisasi kognito: ListDatasets
	sinkronisasi kognito: ListIdentityPoolUsage
	sinkronisasi kognito: ListRecords
	sinkronisasi kognito: RegisterDevice
	sinkronisasi kognito: SetCognitoEvents
	sinkronisasi kognito: SetIdentityPoolConfiguration
	sinkronisasi kognito: SubscribeToDataset
	sinkronisasi kognito: UnsubscribeFromDataset
	sinkronisasi kognito: UpdateRecords

Awalan layanan	Tindakan
memahami-medis	memahami medis: DescribeEntitiesDetectionV2Job memahami medis: DescribeICD10CMInferenceJob memahami medis: DescribePHIDetectionJob memahami medis: DescribeRxNormInferenceJob memahami medis: DescribeSNOMEDCTInferenceJob memahami medis: V2 DetectEntities ComprehendMedical: DetectPHI ComprehendMedical: infericd10cm memahami medis: InferRxNorm ComprehendMedical: InfersnomeDCT memahami medis: ListEntitiesDetectionV2Jobs memahami medis: ListICD10CMInferenceJobs memahami medis: ListPHIDetectionJobs memahami medis: ListRxNormInferenceJobs memahami medis: ListSNOMEDCTInferenceJobs memahami medis: StartEntitiesDetectionV2Job memahami medis: StartICD10CMInferenceJob memahami medis: StartPHIDetectionJob memahami medis: StartRxNormInferenceJob memahami medis: StartSNOMEDCTInferenceJob memahami medis: StopEntitiesDetectionV2Job

Awalan layanan	Tindakan
	memahami medis: StopICD10CMInferenceJob memahami medis: StopPHIDetectionJob memahami medis: StopRxNormInferenceJob memahami medis: StopSNOMEDCTInferenceJob

Awalan layanan	Tindakan
pengoptimal komputasi	pengoptimal komputasi: DeleteRecommendationPreferences
	pengoptimal komputasi: DescribeRecommendationExportJobs
	pengoptimal komputasi: ExportAutoScalingGroupRecommendations
	pengoptimal komputasi: ExportEBSVolumeRecommendations
	pengoptimal komputasi: ExportEC2InstanceRecommendations
	pengoptimal komputasi: ExportECSServiceRecommendations
	pengoptimal komputasi: ExportIdleRecommendations
	pengoptimal komputasi: ExportLambdaFunctionRecommendations
	pengoptimal komputasi: ExportLicenseRecommendations
	pengoptimal komputasi: ExportRDSDatabaseRecommendations
	pengoptimal komputasi: GetEC2RecommendationProjectedMetrics
	pengoptimal komputasi: GetECSServiceRecommendationProjectedMetrics
	pengoptimal komputasi: GetEffectiveRecommendationPreferences
	pengoptimal komputasi: GetEnrollmentStatus
	pengoptimal komputasi: GetEnrollmentStatusesForOrganization
	pengoptimal komputasi: GetRDSDatabaseRecommendationProjectedMetrics
	pengoptimal komputasi: GetRecommendationPreferences
	pengoptimal komputasi: GetRecommendationSummaries
	pengoptimal komputasi: PutRecommendationPreferences

Awalan layanan	Tindakan
	pengoptimal komputasi: UpdateEnrollmentStatus

Awalan layanan	Tindakan
config	konfigurasi: BatchGetResourceConfig konfigurasi: DeleteAggregationAuthorization konfigurasi: DeleteConfigRule konfigurasi: DeleteConfigurationAggregator konfigurasi: DeleteConfigurationRecorder konfigurasi: DeleteConformancePack konfigurasi: DeleteDeliveryChannel konfigurasi: DeleteEvaluationResults konfigurasi: DeleteOrganizationConfigRule konfigurasi: DeleteOrganizationConformancePack konfigurasi: DeletePendingAggregationRequest konfigurasi: DeleteRemediationConfiguration konfigurasi: DeleteRemediationExceptions konfigurasi: DeleteResourceConfig konfigurasi: DeleteRetentionConfiguration konfigurasi: DeleteStoredQuery konfigurasi: DeliverConfigSnapshot konfigurasi: DescribeAggregateComplianceByConfigRules konfigurasi: DescribeAggregateComplianceByConformancePacks konfigurasi: DescribeAggregationAuthorizations konfigurasi: DescribeComplianceByConfigRule

Awalan layanan	Tindakan
	konfigurasi: DescribeComplianceByResource konfigurasi: DescribeConfigRuleEvaluationStatus konfigurasi: DescribeConfigRules konfigurasi: DescribeConfigurationAggregatorSourcesStatus konfigurasi: DescribeConfigurationAggregators konfigurasi: DescribeConfigurationRecorderStatus konfigurasi: DescribeConfigurationRecorders konfigurasi: DescribeConformancePackCompliance konfigurasi: DescribeConformancePackStatus konfigurasi: DescribeConformancePacks konfigurasi: DescribeDeliveryChannelStatus konfigurasi: DescribeDeliveryChannels konfigurasi: DescribeOrganizationConfigRuleStatuses konfigurasi: DescribeOrganizationConfigRules konfigurasi: DescribeOrganizationConformancePackStatuses konfigurasi: DescribeOrganizationConformancePacks konfigurasi: DescribePendingAggregationRequests konfigurasi: DescribeRemediationConfigurations konfigurasi: DescribeRemediationExceptions konfigurasi: DescribeRemediationExecutionStatus konfigurasi: DescribeRetentionConfigurations

Awalan layanan	Tindakan
	konfigurasi: GetComplianceDetailsByConfigRule konfigurasi: GetComplianceDetailsByResource konfigurasi: GetComplianceSummaryByConfigRule konfigurasi: GetComplianceSummaryByResourceType konfigurasi: GetConformancePackComplianceDetails konfigurasi: GetConformancePackComplianceSummary konfigurasi: GetCustomRulePolicy konfigurasi: GetDiscoveredResourceCounts konfigurasi: GetOrganizationConfigRuleDetailedStatus konfigurasi: GetOrganizationConformancePackDetailedStatus konfigurasi: GetOrganizationCustomRulePolicy konfigurasi: GetResourceConfigHistory konfigurasi: GetResourceEvaluationSummary konfigurasi: GetStoredQuery konfigurasi: ListConfigurationRecorders konfigurasi: ListConformancePackComplianceScores konfigurasi: ListConnectors konfigurasi: ListDiscoveredResources konfigurasi: ListResourceEvaluations konfigurasi: ListStoredQueries konfigurasi: PutConfigRule

Awalan layanan	Tindakan
	konfigurasi: PutConfigurationAggregator
	konfigurasi: PutConfigurationRecorder
	konfigurasi: PutConformancePack
	konfigurasi: PutConnector
	konfigurasi: PutDeliveryChannel
	konfigurasi: PutEvaluations
	konfigurasi: PutExternalEvaluation
	konfigurasi: PutOrganizationConfigRule
	konfigurasi: PutOrganizationConformancePack
	konfigurasi: PutRemediationConfigurations
	konfigurasi: PutRemediationExceptions
	konfigurasi: PutResourceConfig
	konfigurasi: PutRetentionConfiguration
	konfigurasi: PutStoredQuery
	konfigurasi: SelectResourceConfig
	konfigurasi: StartConfigRulesEvaluation
	konfigurasi: StartConfigurationRecorder
	konfigurasi: StartRemediationExecution
	konfigurasi: StartResourceEvaluation
	konfigurasi: StopConfigurationRecorder

Awalan layanan	Tindakan
hubungkan	menghubungkan: ActivateEvaluationForm
	menghubungkan: AssociateAnalyticsDataSet
	menghubungkan: AssociateApprovedOrigin
	menghubungkan: AssociateBot
	menghubungkan: AssociateContactWithUser
	menghubungkan: AssociateDefaultVocabulary
	menghubungkan: AssociateEmailAddressAlias
	menghubungkan: AssociateFlow
	menghubungkan: AssociateInstanceStorageConfig
	menghubungkan: AssociateLambdaFunction
	menghubungkan: AssociateLexBot
	menghubungkan: AssociatePhoneNumberContactFlow
	menghubungkan: AssociateQueueEmailAddresses
	menghubungkan: AssociateQueueQuickConnects
	menghubungkan: AssociateRoutingProfileQueues
	menghubungkan: AssociateSecurityKey
	menghubungkan: AssociateUserProficiencies
	menghubungkan: BatchAssociateAnalyticsDataSet
	menghubungkan: BatchCreateDataTableValue
	menghubungkan: BatchDeleteDataTableValue
	menghubungkan: BatchDescribeDataTableValue

Awalan layanan	Tindakan
	menghubungkan: BatchDisassociateAnalyticsDataSet
	menghubungkan: BatchGetFlowAssociation
	menghubungkan: BatchPutContact
	menghubungkan: BatchUpdateDataTableValue
	menghubungkan: ClaimPhoneNumber
	menghubungkan: CreateAgentStatus
	menghubungkan: CreateAttachedFile
	menghubungkan: CreateContact
	menghubungkan: CreateContactFlow
	menghubungkan: CreateContactFlowModule
	menghubungkan: CreateContactFlowModuleAlias
	menghubungkan: CreateContactFlowModuleVersion
	menghubungkan: CreateContactFlowVersion
	menghubungkan: CreateDataTable
	menghubungkan: CreateDataTableAttribute
	menghubungkan: CreateEmailAddress
	menghubungkan: CreateEvaluationForm
	menghubungkan: CreateHoursOfOperation
	menghubungkan: CreateInstance
	menghubungkan: CreateIntegrationAssociation
	menghubungkan: CreateParticipant

Awalan layanan	Tindakan
	menghubungkan: CreatePersistentContactAssociation
	menghubungkan: CreatePredefinedAttribute
	menghubungkan: CreatePrompt
	menghubungkan: CreatePushNotificationRegistration
	menghubungkan: CreateQueue
	menghubungkan: CreateQuickConnect
	menghubungkan: CreateRoutingProfile
	menghubungkan: CreateRule
	menghubungkan: CreateSecurityProfile
	menghubungkan: CreateTaskTemplate
	menghubungkan: CreateTrafficDistributionGroup
	menghubungkan: CreateUseCase
	menghubungkan: CreateUser
	menghubungkan: CreateUserHierarchyGroup
	menghubungkan: CreateView
	menghubungkan: CreateViewVersion
	menghubungkan: CreateVocabulary
	menghubungkan: CreateWorkspace
	menghubungkan: DeactivateEvaluationForm
	menghubungkan: DeleteContactEvaluation
	menghubungkan: DeleteContactFlow

Awalan layanan	Tindakan
	menghubungkan: DeleteContactFlowModule
	menghubungkan: DeleteContactFlowModuleAlias
	menghubungkan: DeleteContactFlowModuleVersion
	menghubungkan: DeleteContactFlowVersion
	menghubungkan: DeleteDataTable
	menghubungkan: DeleteDataTableAttribute
	menghubungkan: DeleteEmailAddress
	menghubungkan: DeleteEvaluationForm
	menghubungkan: DeleteHoursOfOperation
	menghubungkan: DeleteHoursOfOperationOverride
	menghubungkan: DeleteInstance
	menghubungkan: DeleteIntegrationAssociation
	menghubungkan: DeleteNotification
	menghubungkan: DeletePredefinedAttribute
	menghubungkan: DeletePrompt
	menghubungkan: DeletePushNotificationRegistration
	menghubungkan: DeleteQueue
	menghubungkan: DeleteQuickConnect
	menghubungkan: DeleteRoutingProfile
	menghubungkan: DeleteRule
	menghubungkan: DeleteSecurityProfile

Awalan layanan	Tindakan
	menghubungkan: DeleteTaskTemplate
	menghubungkan: DeleteTrafficDistributionGroup
	menghubungkan: DeleteUseCase
	menghubungkan: DeleteUser
	menghubungkan: DeleteUserHierarchyGroup
	menghubungkan: DeleteView
	menghubungkan: DeleteVocabulary
	menghubungkan: DeleteWorkspace
	menghubungkan: DeleteWorkspaceMedia
	menghubungkan: DescribeAttachedFilesConfiguration
	menghubungkan: DescribeAuthenticationProfile
	menghubungkan: DescribeContactFlowModuleAlias
	menghubungkan: DescribeDataTableAttribute
	menghubungkan: DescribeHoursOfOperationOverride
	menghubungkan: DescribeInstanceAttribute
	menghubungkan: DescribeInstanceStorageConfig
	menghubungkan: DescribeNotification
	menghubungkan: DescribePhoneNumber
	menghubungkan: DescribeRule
	menghubungkan: DescribeTrafficDistributionGroup
	menghubungkan: DescribeUserHierarchyStructure

Awalan layanan	Tindakan
	menghubungkan: DescribeVocabulary
	menghubungkan: DisassociateAnalyticsDataSet
	menghubungkan: DisassociateApprovedOrigin
	menghubungkan: DisassociateBot
	menghubungkan: DisassociateEmailAddressAlias
	menghubungkan: DisassociateFlow
	menghubungkan: DisassociateInstanceStorageConfig
	menghubungkan: DisassociateLambdaFunction
	menghubungkan: DisassociateLexBot
	menghubungkan: DisassociatePhoneNumberContactFlow
	menghubungkan: DisassociateQueueEmailAddresses
	menghubungkan: DisassociateQueueQuickConnects
	menghubungkan: DisassociateRoutingProfileQueues
	menghubungkan: DisassociateSecurityKey
	menghubungkan: DisassociateUserProficiencies
	menghubungkan: DismissUserContact
	menghubungkan: EvaluateDataTableValues
	menghubungkan: GetContactAttributes
	menghubungkan: GetContactMetrics
	menghubungkan: GetCurrentMetricData
	menghubungkan: GetCurrentUserData

Awalan layanan	Tindakan
	menghubungkan: GetEffectiveHoursOfOperations
	menghubungkan: GetEvaluationFormValidation
	menghubungkan: GetFederationToken
	menghubungkan: GetFlowAssociation
	menghubungkan: GetMetricData
	menghubungkan: GetMetricDataV2
	menghubungkan: GetPromptFile
	menghubungkan: GetTaskTemplate
	menghubungkan: GetTrafficDistribution
	menghubungkan: ImportPhoneNumber
	menghubungkan: ImportWorkspaceMedia
	menghubungkan: ListAnalyticsDataAssociations
	menghubungkan: ListAnalyticsDataLakeDataSets
	menghubungkan: ListApprovedOrigins
	menghubungkan: ListAssociatedContacts
	menghubungkan: ListAttachedFilesConfigurations
	menghubungkan: ListAuthenticationProfiles
	menghubungkan: ListBots
	menghubungkan: ListContactEvaluations
	menghubungkan: ListContactFlowModuleAliases
	menghubungkan: ListContactFlowModuleVersions

Awalan layanan	Tindakan
	menghubungkan: ListContactFlowModules
	menghubungkan: ListContactFlowVersions
	menghubungkan: ListContactFlows
	menghubungkan: ListContactReferences
	menghubungkan: ListDataTableAttributes
	menghubungkan: ListDataTablePrimaryValues
	menghubungkan: ListDataTableValues
	menghubungkan: ListDataTables
	menghubungkan: ListDefaultVocabularies
	menghubungkan: ListEvaluationFormVersions
	menghubungkan: ListEvaluationForms
	menghubungkan: ListFlowAssociations
	menghubungkan: ListHoursOfOperations
	menghubungkan: ListInstanceAttributes
	menghubungkan: ListInstanceStorageConfigs
	menghubungkan: ListIntegrationAssociations
	menghubungkan: ListLambdaFunctions
	menghubungkan: ListLexBots
	menghubungkan: ListNotifications
	menghubungkan: ListPhoneNumbers
	menghubungkan: ListPhoneNumbersV2

Awalan layanan	Tindakan
	menghubungkan: ListPredefinedAttributes
	menghubungkan: ListPrompts
	menghubungkan: ListQueueEmailAddresses
	menghubungkan: ListQueueQuickConnects
	menghubungkan: ListQueues
	menghubungkan: ListQuickConnects
	menghubungkan: ListRealtimeContactAnalysisSegmentsV2
	menghubungkan: ListRoutingProfileManualAssignmentQueues
	menghubungkan: ListRoutingProfileQueues
	menghubungkan: ListRoutingProfiles
	menghubungkan: ListRules
	menghubungkan: ListSecurityKeys
	menghubungkan: ListSecurityProfileApplications
	menghubungkan: ListSecurityProfileFlowModules
	menghubungkan: ListSecurityProfilePermissions
	menghubungkan: ListSecurityProfiles
	menghubungkan: ListTaskTemplates
	menghubungkan: ListTrafficDistributionGroups
	menghubungkan: ListUseCases
	menghubungkan: ListUserHierarchyGroups
	menghubungkan: ListUserNotifications

Awalan layanan	Tindakan
	menghubungkan: ListUsers
	menghubungkan: ListViewVersions
	menghubungkan: ListViews
	menghubungkan: ListWorkspaceMedia
	menghubungkan: ListWorkspacePages
	menghubungkan: ListWorkspaces
	menghubungkan: MonitorContact
	menghubungkan: PauseContact
	menghubungkan: PutUserStatus
	menghubungkan: ReleasePhoneNumber
	menghubungkan: ReplicateInstance
	menghubungkan: ResumeContact
	menghubungkan: ResumeContactRecording
	menghubungkan: SearchAgentStatuses
	menghubungkan: SearchAvailablePhoneNumbers
	menghubungkan: SearchContactEvaluations
	menghubungkan: SearchContactFlowModules
	menghubungkan: SearchContactFlows
	menghubungkan: SearchContacts
	menghubungkan: SearchDataTables
	menghubungkan: SearchEmailAddresses

Awalan layanan	Tindakan
	menghubungkan: SearchEvaluationForms
	menghubungkan: SearchHoursOfOperations
	menghubungkan: SearchPredefinedAttributes
	menghubungkan: SearchPrompts
	menghubungkan: SearchQueues
	menghubungkan: SearchQuickConnects
	menghubungkan: SearchRoutingProfiles
	menghubungkan: SearchRules
	menghubungkan: SearchSecurityProfiles
	menghubungkan: SearchUserHierarchyGroups
	menghubungkan: SearchViews
	menghubungkan: SearchVocabularies
	menghubungkan: SearchWorkspaceAssociations
	menghubungkan: SearchWorkspaces
	menghubungkan: SendChatIntegrationEvent
	menghubungkan: SendOutboundEmail
	menghubungkan: StartChatContact
	menghubungkan: StartContactConversationalAnalyticsJob
	menghubungkan: StartContactEvaluation
	menghubungkan: StartContactMediaProcessing
	menghubungkan: StartContactRecording

Awalan layanan	Tindakan
	menghubungkan: StartContactStreaming
	menghubungkan: StartEmailContact
	menghubungkan: StartOutboundChatContact
	menghubungkan: StartOutboundEmailContact
	menghubungkan: StartOutboundVoiceContact
	menghubungkan: StartScreenSharing
	menghubungkan: StartTaskContact
	menghubungkan: StartWebRTCContact
	menghubungkan: StopContact
	menghubungkan: StopContactMediaProcessing
	menghubungkan: StopContactRecording
	menghubungkan: StopContactStreaming
	menghubungkan: SubmitContactEvaluation
	menghubungkan: SuspendContactRecording
	menghubungkan: TransferContact
	menghubungkan: UpdateAgentStatus
	menghubungkan: UpdateAttachedFilesConfiguration
	menghubungkan: UpdateAuthenticationProfile
	menghubungkan: UpdateContact
	menghubungkan: UpdateContactAttributes
	menghubungkan: UpdateContactEvaluation

Awalan layanan	Tindakan
	menghubungkan: UpdateContactFlowContent
	menghubungkan: UpdateContactFlowMetadata
	menghubungkan: UpdateContactFlowModuleAlias
	menghubungkan: UpdateContactFlowModuleContent
	menghubungkan: UpdateContactFlowModuleMetadata
	menghubungkan: UpdateContactFlowName
	menghubungkan: UpdateContactRoutingData
	menghubungkan: UpdateContactSchedule
	menghubungkan: UpdateDataTableAttribute
	menghubungkan: UpdateDataTableMetadata
	menghubungkan: UpdateDataTablePrimaryValues
	menghubungkan: UpdateEmailAddressMetadata
	menghubungkan: UpdateEvaluationForm
	menghubungkan: UpdateHoursOfOperation
	menghubungkan: UpdateHoursOfOperationOverride
	menghubungkan: UpdateInstanceAttribute
	menghubungkan: UpdateInstanceStorageConfig
	menghubungkan: UpdateNotificationContent
	menghubungkan: UpdateParticipantAuthentication
	menghubungkan: UpdateParticipantRoleConfig
	menghubungkan: UpdatePhoneNumber

Awalan layanan	Tindakan
	menghubungkan: UpdatePhoneNumberMetadata
	menghubungkan: UpdatePredefinedAttribute
	menghubungkan: UpdatePrompt
	menghubungkan: UpdateQueueHoursOfOperation
	menghubungkan: UpdateQueueMaxContacts
	menghubungkan: UpdateQueueName
	menghubungkan: UpdateQueueOutboundCallerConfig
	menghubungkan: UpdateQueueOutboundEmailConfig
	menghubungkan: UpdateQueueStatus
	menghubungkan: UpdateQuickConnectConfig
	menghubungkan: UpdateQuickConnectName
	menghubungkan: UpdateRoutingProfileAgentAvailabilityTimer
	menghubungkan: UpdateRoutingProfileConcurrency
	menghubungkan: UpdateRoutingProfileDefaultOutboundQueue
	menghubungkan: UpdateRoutingProfileName
	menghubungkan: UpdateRoutingProfileQueues
	menghubungkan: UpdateRule
	menghubungkan: UpdateSecurityProfile
	menghubungkan: UpdateTaskTemplate
	menghubungkan: UpdateTrafficDistribution
	menghubungkan: UpdateUserHierarchy

Awalan layanan	Tindakan
	menghubungkan: UpdateUserHierarchyGroupName menghubungkan: UpdateUserHierarchyStructure menghubungkan: UpdateUserIdentityInfo menghubungkan: UpdateUserNotificationStatus menghubungkan: UpdateUserPhoneConfig menghubungkan: UpdateUserProficiencies menghubungkan: UpdateUserRoutingProfile menghubungkan: UpdateUserSecurityProfiles menghubungkan: UpdateViewContent menghubungkan: UpdateViewMetadata menghubungkan: UpdateWorkspaceMetadata menghubungkan: UpdateWorkspaceTheme menghubungkan: UpdateWorkspaceVisibility
cur	kelebar: DeleteReportDefinition kelebar: DescribeReportDefinitions kelebar: ModifyReportDefinition kelebar: PutReportDefinition

Awalan layanan	Tindakan
databrew	Databrew: BatchDeleteRecipeVersion
	Databrew: CreateDataset
	Databrew: CreateProfileJob
	Databrew: CreateProject
	Databrew: CreateRecipe
	Databrew: CreateRecipeJob
	Databrew: CreateRuleset
	Databrew: CreateSchedule
	Databrew: DeleteDataset
	Databrew: DeleteJob
	Databrew: DeleteProject
	Databrew: DeleteRecipeVersion
	Databrew: DeleteRuleset
	Databrew: DeleteSchedule
	Databrew: DescribeDataset
	Databrew: DescribeJob
	Databrew: DescribeJobRun
	Databrew: DescribeProject
	Databrew: DescribeRecipe
	Databrew: DescribeRuleset
	Databrew: DescribeSchedule

Awalan layanan	Tindakan
	Databrew: ListDatasets
	Databrew: ListJobRuns
	Databrew: ListJobs
	Databrew: ListProjects
	Databrew: ListRecipeVersions
	Databrew: ListRecipes
	Databrew: ListRulesets
	Databrew: ListSchedules
	Databrew: PublishRecipe
	Databrew: SendProjectSessionAction
	Databrew: StartJobRun
	Databrew: StartProjectSession
	Databrew: StopJobRun
	Databrew: UpdateDataset
	Databrew: UpdateProfileJob
	Databrew: UpdateProject
	database: UpdateRecipe
	database: UpdateRecipeJob
	database: UpdateRuleset
	database: UpdateSchedule

Awalan layanan	Tindakan
pertukaran data	pertukaran data: AcceptDataGrant pertukaran data: CancelJob pertukaran data: CreateDataGrant pertukaran data: CreateDataSet pertukaran data: CreateEventAction pertukaran data: CreateJob pertukaran data: CreateRevision pertukaran data: DeleteAsset pertukaran data: DeleteDataGrant pertukaran data: DeleteEventAction pertukaran data: DeleteRevision pertukaran data: GetDataGrant pertukaran data: GetEventAction pertukaran data: GetJob pertukaran data: GetReceivedDataGrant pertukaran data: ListDataGrants pertukaran data: ListDataSetRevisions pertukaran data: ListDataSets pertukaran data: ListEventActions pertukaran data: ListJobs pertukaran data: ListReceivedDataGrants

Awalan layanan	Tindakan
	pertukaran data: ListRevisionAssets pertukaran data: RevokeRevision pertukaran data: SendDataSetNotification pertukaran data: StartJob pertukaran data: UpdateAsset pertukaran data: UpdateDataSet pertukaran data: UpdateEventAction pertukaran data: UpdateRevision

Awalan layanan	Tindakan
datapipeline	datapipeline: ActivatePipeline
	datapipeline: CreatePipeline
	datapipeline: DeactivatePipeline
	datapipeline: DeletePipeline
	datapipeline: DescribeObjects
	datapipeline: DescribePipelines
	datapipeline: EvaluateExpression
	datapipeline: GetPipelineDefinition
	datapipeline: ListPipelines
	datapipeline: PollForTask
	datapipeline: PutPipelineDefinition
	datapipeline: QueryObjects
	datapipeline: ReportTaskProgress
	datapipeline: ReportTaskRunnerHeartbeat
	datapipeline: SetStatus
	datapipeline: SetTaskStatus
	datapipeline: ValidatePipelineDefinition

Awalan layanan	Tindakan
dax	dax: CreateCluster dax: DecreaseReplicationFactor dax: DeleteCluster dax: DeleteParameterGroup dax: DeleteSubnetGroup dax: DescribeClusters dax: DescribeDefaultParameters dax: DescribeEvents dax: DescribeParameterGroups dax: DescribeParameters dax: DescribeSubnetGroups dax: IncreaseReplicationFactor dax: RebootNode dax: UpdateCluster dax: UpdateParameterGroup dax: UpdateSubnetGroup

Awalan layanan	Tindakan
devicefarm	devicefarm: CreateDevicePool
	devicefarm: CreateInstanceProfile
	devicefarm: CreateNetworkProfile
	devicefarm: CreateProject
	devicefarm: CreateRemoteAccessSession
	devicefarm: CreateTestGridProject
	devicefarm: CreateTestGridUrl
	devicefarm: CreateUpload
	DeviceFarm:createVPCEConfiguration
	devicefarm: DeleteDevicePool
	devicefarm: DeleteInstanceProfile
	devicefarm: DeleteNetworkProfile
	devicefarm: DeleteProject
	devicefarm: DeleteRemoteAccessSession
	devicefarm: DeleteRun
	devicefarm: DeleteTestGridProject
	devicefarm: DeleteUpload
	DeviceFarm:deletevpceconfiguration
	devicefarm: GetAccountSettings
	devicefarm: GetDevice
	devicefarm: GetDeviceInstance

Awalan layanan	Tindakan
	devicefarm: GetDevicePool
	devicefarm: GetDevicePoolCompatibility
	devicefarm: GetInstanceProfile
	devicefarm: GetJob
	devicefarm: GetNetworkProfile
	devicefarm: GetOfferingStatus
	devicefarm: GetProject
	devicefarm: GetRemoteAccessSession
	devicefarm: GetRun
	devicefarm: GetSuite
	devicefarm: GetTest
	devicefarm: GetTestGridProject
	devicefarm: GetTestGridSession
	devicefarm: GetUpload
	DeviceFarm:GetVPCEConfiguration
	devicefarm: ListArtifacts
	devicefarm: ListDeviceInstances
	devicefarm: ListDevicePools
	devicefarm: ListDevices
	devicefarm: ListInstanceProfiles
	devicefarm: ListJobs

Awalan layanan	Tindakan
	devicefarm: ListNetworkProfiles
	devicefarm: ListOfferingPromotions
	devicefarm: ListOfferingTransactions
	devicefarm: ListOfferings
	devicefarm: ListProjects
	devicefarm: ListRemoteAccessSessions
	devicefarm: ListRuns
	devicefarm: ListSamples
	devicefarm: ListSuites
	devicefarm: ListTestGridProjects
	devicefarm: ListTestGridSessionActions
	devicefarm: ListTestGridSessionArtifacts
	devicefarm: ListTestGridSessions
	devicefarm: ListTests
	devicefarm: ListUniqueProblems
	devicefarm: ListUploads
	DeviceFarm:ListVPCEConfigurations
	devicefarm: PurchaseOffering
	devicefarm: RenewOffering
	devicefarm: ScheduleRun
	devicefarm: StopJob

Awalan layanan	Tindakan
	devicefarm: StopRemoteAccessSession
	devicefarm: StopRun
	devicefarm: UpdateDeviceInstance
	devicefarm: UpdateDevicePool
	devicefarm: UpdateInstanceProfile
	devicefarm: UpdateNetworkProfile
	devicefarm: UpdateProject
	devicefarm: UpdateTestGridProject
	devicefarm: UpdateUpload
	DeviceFarm:updatevpceconfiguration

Awalan layanan	Tindakan
devops-guru	devops-guru: AddNotificationChannel devops-guru: DeleteInsight devops-guru: DescribeAccountHealth devops-guru: DescribeAccountOverview devops-guru: DescribeAnomaly devops-guru: DescribeEventSourcesConfig devops-guru: DescribeFeedback devops-guru: DescribeInsight devops-guru: DescribeOrganizationHealth devops-guru: DescribeOrganizationOverview devops-guru: DescribeOrganizationResourceCollectionHealth devops-guru: DescribeResourceCollectionHealth devops-guru: DescribeServiceIntegration devops-guru: GetCostEstimation devops-guru: GetResourceCollection devops-guru: ListAnomaliesForInsight devops-guru: ListAnomalousLogGroups devops-guru: ListEvents devops-guru: ListInsights devops-guru: ListMonitoredResources devops-guru: ListNotificationChannels

Awalan layanan	Tindakan
	devops-guru: ListOrganizationInsights devops-guru: ListRecommendations devops-guru: PutFeedback devops-guru: RemoveNotificationChannel devops-guru: SearchInsights devops-guru: SearchOrganizationInsights devops-guru: StartCostEstimation devops-guru: UpdateEventSourcesConfig devops-guru: UpdateResourceCollection devops-guru: UpdateServiceIntegration

Awalan layanan	Tindakan
DirectConnect	directconnect: AcceptDirectConnectGatewayAssociationProposal
	directconnect: AllocateConnectionOnInterconnect
	directconnect: AllocateHostedConnection
	directconnect: AllocatePrivateVirtualInterface
	directconnect: AllocatePublicVirtualInterface
	directconnect: AllocateTransitVirtualInterface
	directconnect: AssociateConnectionWithLag
	directconnect: AssociateHostedConnection
	directconnect: AssociateMacSecKey
	directconnect: AssociateVirtualInterface
	directconnect: ConfirmConnection
	directconnect: ConfirmCustomerAgreement
	directconnect: ConfirmPrivateVirtualInterface
	directconnect: ConfirmPublicVirtualInterface
	directconnect: ConfirmTransitVirtualInterface
	DirectConnect:CreateBGPpeer
	directconnect: CreateConnection
	directconnect: CreateDirectConnectGateway
	directconnect: CreateDirectConnectGatewayAssociation
	directconnect: CreateDirectConnectGatewayAssociationProposal
	directconnect: CreateInterconnect

Awalan layanan	Tindakan
	directconnect: CreateLag
	directconnect: CreatePrivateVirtualInterface
	directconnect: CreatePublicVirtualInterface
	directconnect: CreateTransitVirtualInterface
	DirectConnect:DeleteBGPPeer
	directconnect: DeleteConnection
	directconnect: DeleteDirectConnectGateway
	directconnect: DeleteDirectConnectGatewayAssociation
	directconnect: DeleteDirectConnectGatewayAssociationProposal
	directconnect: DeleteInterconnect
	directconnect: DeleteLag
	directconnect: DeleteVirtualInterface
	directconnect: DescribeConnectionLoa
	directconnect: DescribeConnections
	directconnect: DescribeConnectionsOnInterconnect
	directconnect: DescribeCustomerMetadata
	directconnect: DescribeDirectConnectGatewayAssociationProposals
	directconnect: DescribeDirectConnectGatewayAssociations
	directconnect: DescribeDirectConnectGatewayAttachments
	directconnect: DescribeDirectConnectGateways

Awalan layanan	Tindakan
	directconnect: DescribeHostedConnections
	directconnect: DescribeInterconnectLoa
	directconnect: DescribeInterconnects
	directconnect: DescribeLags
	directconnect: DescribeLoa
	directconnect: DescribeLocations
	directconnect: DescribeRouterConfiguration
	directconnect: DescribeVirtualGateways
	directconnect: DescribeVirtualInterfaces
	directconnect: DisassociateConnectionFromLag
	directconnect: DisassociateMacSecKey
	directconnect: ListVirtualInterfaceTestHistory
	directconnect: StartBgpFailoverTest
	directconnect: StopBgpFailoverTest
	directconnect: UpdateConnection
	directconnect: UpdateDirectConnectGateway
	directconnect: UpdateDirectConnectGatewayAssociation
	directconnect: UpdateLag
	directconnect: UpdateVirtualInterfaceAttributes

Awalan layanan	Tindakan
dlm	dlm: CreateLifecyclePolicy dlm: DeleteLifecyclePolicy dlm: GetLifecyclePolicies dlm: GetLifecyclePolicy dlm: UpdateLifecyclePolicy

Awalan layanan	Tindakan
dms	dms: ApplyPendingMaintenanceAction
	dms: AssociateExtensionPack
	dms: BatchStartRecommendations
	dms: CancelMetadataModelCreation
	dms: CancelReplicationTaskAssessmentRun
	dms: CreateDataProvider
	dms: CreateEndpoint
	dms: CreateEventSubscription
	dms: CreateInstanceProfile
	dms: CreateMigrationProject
	dms: CreateReplicationConfig
	dms: CreateReplicationInstance
	dms: CreateReplicationSubnetGroup
	dms: CreateReplicationTask
	dms: DeleteCertificate
	dms: DeleteConnection
	dms: DeleteDataMigration
	dms: DeleteDataProvider
	dms: DeleteEndpoint
	dms: DeleteEventSubscription
	dms: DeleteFleetAdvisorCollector

Awalan layanan	Tindakan
	dms: DeleteFleetAdvisorDatabases
	dms: DeleteInstanceProfile
	dms: DeleteMigrationProject
	dms: DeleteReplicationConfig
	dms: DeleteReplicationInstance
	dms: DeleteReplicationSubnetGroup
	dms: DeleteReplicationTask
	dms: DeleteReplicationTaskAssessmentRun
	dms: DescribeAccountAttributes
	dms: DescribeApplicableIndividualAssessments
	dms: DescribeCertificates
	dms: DescribeConnections
	dms: DescribeDataMigrations
	dms: DescribeEndpointSettings
	dms: DescribeEndpointTypes
	dms: DescribeEndpoints
	dms: DescribeEngineVersions
	dms: DescribeEventCategories
	dms: DescribeEventSubscriptions
	dms: DescribeEvents
	dms: DescribeFleetAdvisorCollectors

Awalan layanan	Tindakan
	dms: DescribeFleetAdvisorDatabases
	dms: DescribeFleetAdvisorLsaAnalysis
	dms: DescribeFleetAdvisorSchemaObjectSummary
	dms: DescribeFleetAdvisorSchemas
	dms: DescribeMetadataModel
	dms: DescribeMetadataModelChildren
	dms: DescribeMetadataModelCreations
	dms: DescribeMetadataModelImports
	dms: DescribeOrderableReplicationInstances
	dms: DescribePendingMaintenanceActions
	dms: DescribeRecommendationLimitations
	dms: DescribeRecommendations
	dms: DescribeRefreshSchemasStatus
	dms: DescribeReplicationConfigs
	dms: DescribeReplicationInstanceTaskLogs
	dms: DescribeReplicationInstances
	dms: DescribeReplicationSubnetGroups
	dms: DescribeReplicationTableStatistics
	dms: DescribeReplicationTaskAssessmentResults
	dms: DescribeReplicationTaskAssessmentRuns
	dms: DescribeReplicationTaskIndividualAssessments

Awalan layanan	Tindakan
	dms: DescribeReplicationTasks
	dms: DescribeReplications
	dms: DescribeSchemas
	dms: DescribeTableStatistics
	dms: ExportMetadataModelAssessment
	dms: ImportCertificate
	dms: ListDataProviders
	dms: ListExtensionPacks
	dms: ListInstanceProfiles
	dms: ListMetadataModelAssessments
	dms: ListMetadataModelConversions
	dms: ListMetadataModelExports
	dms: ListMigrationProjects
	dms: ModifyDataMigration
	dms: ModifyEndpoint
	dms: ModifyEventSubscription
	dms: ModifyReplicationConfig
	dms: ModifyReplicationInstance
	dms: ModifyReplicationSubnetGroup
	dms: ModifyReplicationTask
	dms: MoveReplicationTask

Awalan layanan	Tindakan
	dms: RebootReplicationInstance
	dms: RefreshSchemas
	dms: ReloadReplicationTables
	dms: ReloadTables
	dms: RunFleetAdvisorLsaAnalysis
	dms: StartMetadataModelAssessment
	dms: StartMetadataModelConversion
	dms: StartMetadataModelCreation
	dms: StartMetadataModelExportAsScripts
	dms: StartMetadataModelExportToTarget
	dms: StartRecommendations
	dms: StartReplication
	dms: StartReplicationTask
	dms: StartReplicationTaskAssessment
	dms: StopDataMigration
	dms: StopReplicationTask
	dms: TestConnection
	dms: UpdateConversionConfiguration
	dms: UpdateDataProvider
	dms: UpdateInstanceProfile
	dms: UpdateMigrationProject

Awalan layanan	Tindakan
	dms: UpdateSubscriptionsToEventBridge
docdb-elastis	docdb-elastis: ApplyPendingMaintenanceAction docdb-elastis: CopyClusterSnapshot docdb-elastis: DeleteCluster docdb-elastis: DeleteClusterSnapshot docdb-elastis: GetCluster docdb-elastis: GetClusterSnapshot docdb-elastis: GetPendingMaintenanceAction docdb-elastis: ListClusterSnapshots docdb-elastis: ListClusters docdb-elastis: ListPendingMaintenanceActions docdb-elastis: RestoreClusterFromSnapshot docdb-elastis: StartCluster docdb-elastis: StopCluster docdb-elastis: UpdateCluster

Awalan layanan	Tindakan
dynamodb	dinamodb: CreateBackup
	dinamodb: CreateGlobalTable
	dinamodb: CreateTable
	dinamodb: DeleteBackup
	dinamodb: DeleteTable
	dinamodb: DescribeBackup
	dinamodb: DescribeContinuousBackups
	dinamodb: DescribeContributorInsights
	dinamodb: DescribeEndpoints
	dinamodb: DescribeExport
	dinamodb: DescribeGlobalTable
	dinamodb: DescribeGlobalTableSettings
	dinamodb: DescribeImport
	dinamodb: DescribeKinesisStreamingDestination
	dinamodb: DescribeLimits
	dinamodb: DescribeStream
	dinamodb: DescribeTable
	dinamodb: DescribeTableReplicaAutoScaling
	dinamodb: DescribeTimeToLive
	dinamodb: DisableKinesisStreamingDestination
	dinamodb: EnableKinesisStreamingDestination

Awalan layanan	Tindakan
	dinamodb: ExportTableToPointInTime
	dinamodb: GetResourcePolicy
	dinamodb: ImportTable
	dinamodb: ListBackups
	dinamodb: ListContributorInsights
	dinamodb: ListExports
	dinamodb: ListGlobalTables
	dinamodb: ListImports
	dinamodb: ListStreams
	dinamodb: ListTables
	dinamodb: RestoreTableFromBackup
	dinamodb: RestoreTableToPointInTime
	dinamodb: UpdateContinuousBackups
	dinamodb: UpdateContributorInsights
	dinamodb: UpdateGlobalTable
	dinamodb: UpdateGlobalTableSettings
	dinamodb: UpdateKinesisStreamingDestination
	dinamodb: UpdateTable
	dinamodb: UpdateTableReplicaAutoScaling
	dinamodb: UpdateTimeToLive

Awalan layanan	Tindakan
ebs	ebs: CompleteSnapshot ebs: StartSnapshot

Awalan layanan	Tindakan
ec2	EC2: AcceptAddressTransfer
	EC2: AcceptCapacityReservationBillingOwnership
	EC2: AcceptReservedInstancesExchangeQuote
	EC2: AcceptTransitGatewayClientVpnAttachment
	EC2: AcceptTransitGatewayMulticastDomainAssociations
	EC2: AcceptTransitGatewayPeeringAttachment
	EC2: AcceptTransitGatewayVpcAttachment
	EC2: AcceptVpcEndpointConnections
	EC2: AcceptVpcPeeringConnection
	EC2: AdvertiseByoipCidr
	EC2: AllocateAddress
	EC2: AllocateHosts
	EC2: AllocateIpamPoolCidr
	EC2: ApplySecurityGroupsToClientVpnTargetNetwork
	EC2: AssignIpv6Addresses
	EC2: AssignPrivateIpAddresses
	EC2: AssignPrivateNatGatewayAddress
	EC2: AssociateAddress
	EC2: AssociateCapacityReservationBillingOwner
	EC2: AssociateClientVpnTargetNetwork
	EC2: AssociateDhcpOptions

Awalan layanan	Tindakan
	EC2: AssociateEnclaveCertificateIamRole
	EC2: AssociateIamInstanceProfile
	EC2: AssociateInstanceEventWindow
	EC2: AssociateIpamByoasn
	EC2: AssociateIpamResourceDiscovery
	EC2: AssociateNatGatewayAddress
	EC2: AssociateRouteServer
	EC2: AssociateRouteTable
	EC2: AssociateSecurityGroupVpc
	EC2: AssociateSubnetCidrBlock
	EC2: AssociateTransitGatewayMulticastDomain
	EC2: AssociateTransitGatewayPolicyTable
	EC2: AssociateTransitGatewayRouteTable
	EC2: AssociateTrunkInterface
	EC2: AssociateVpcCidrBlock
	EC2: AttachClassicLinkVpc
	EC2: AttachImageWatermark
	EC2: AttachInternetGateway
	EC2: AttachNetworkInterface
	EC2: AttachVerifiedAccessTrustProvider
	EC2: AttachVolume

Awalan layanan	Tindakan
	EC2: AttachVpnGateway
	EC2: AuthorizeClientVpnIngress
	EC2: AuthorizeSecurityGroupEgress
	EC2: AuthorizeSecurityGroupIngress
	EC2: BundleInstance
	EC2: CancelBundleTask
	EC2: CancelCapacityReservation
	EC2: CancelCapacityReservationFleets
	EC2: CancelConversionTask
	EC2: CancelDeclarativePoliciesReport
	EC2: CancelExportTask
	EC2: CancelImageLaunchPermission
	EC2: CancelImportTask
	EC2: CancelReservedInstancesListing
	EC2: CancelSpotFleetRequests
	EC2: CancelSpotInstanceRequests
	EC2: ConfirmProductInstance
	EC2: CopyFpgaImage
	EC2: CopyImage
	EC2: CopySnapshot
	EC2: CopyVolumes

Awalan layanan	Tindakan
	EC2: CreateCapacityManagerDataExport
	EC2: CreateCapacityReservation
	EC2: CreateCapacityReservationBySplitting
	EC2: CreateCapacityReservationCancellationQuote
	EC2: CreateCapacityReservationFleet
	EC2: CreateCarrierGateway
	EC2: CreateClientVpnEndpoint
	EC2: CreateClientVpnRoute
	EC2: CreateCoipCidr
	EC2: CreateCoipPool
	EC2: CreateCustomerGateway
	EC2: CreateDefaultSubnet
	EC2: CreateDefaultVpc
	EC2: CreateDelegateMacVolumeOwnershipTask
	EC2: CreateDhcpOptions
	EC2: CreateEgressOnlyInternetGateway
	EC2: CreateFleet
	EC2: CreateFlowLogs
	EC2: CreateFpgaImage
	EC2: CreateImage
	EC2: CreateImageUsageReport

Awalan layanan	Tindakan
	EC2: CreateInstanceConnectEndpoint
	EC2: CreateInstanceEventWindow
	EC2: CreateInstanceExportTask
	EC2: CreateInternetGateway
	EC2: CreateInterruptibleCapacityReservationAllocation
	EC2: CreateIpam
	EC2: CreateIpamExternalResourceVerificationToken
	EC2: CreateIpamPolicy
	EC2: CreateIpamPool
	EC2: CreateIpamPrefixListResolver
	EC2: CreateIpamPrefixListResolverTarget
	EC2: CreateIpamResourceDiscovery
	EC2: CreateIpamScope
	EC2: CreateKeyPair
	EC2: CreateLaunchTemplateVersion
	EC2: CreateLocalGatewayRoute
	EC2: CreateLocalGatewayRouteTable
	EC2: CreateLocalGatewayRouteTableVirtualInterfaceGroupAssoc iation
	EC2: CreateLocalGatewayRouteTableVpcAssociation
	EC2: CreateLocalGatewayVirtualInterface

Awalan layanan	Tindakan
	EC2: CreateLocalGatewayVirtualInterfaceGroup
	EC2: CreateMacSystemIntegrityProtectionModificationTask
	EC2: CreateManagedPrefixList
	EC2: CreateNatGateway
	EC2: CreateNetworkAcl
	EC2: CreateNetworkAclEntry
	EC2: CreateNetworkInsightsAccessScope
	EC2: CreateNetworkInsightsPath
	EC2: CreateNetworkInterface
	EC2: CreateNetworkInterfacePermission
	EC2: CreatePlacementGroup
	EC2: CreatePublicIpv4Pool
	EC2: CreateReplaceRootVolumeTask
	EC2: CreateReservedInstancesListing
	EC2: CreateRestoreImageTask
	EC2: CreateRoute
	EC2: CreateRouteServer
	EC2: CreateRouteServerEndpoint
	EC2: CreateRouteServerPeer
	EC2: CreateRouteTable
	EC2: CreateSecondaryNetwork

Awalan layanan	Tindakan
	EC2: CreateSecondarySubnet
	EC2: CreateSecurityGroup
	EC2: CreateSnapshots
	EC2: CreateSpotDatafeedSubscription
	EC2: CreateStoreImageTask
	EC2: CreateSubnet
	EC2: CreateSubnetCidrReservation
	EC2: CreateTrafficMirrorFilter
	EC2: CreateTrafficMirrorFilterRule
	EC2: CreateTrafficMirrorSession
	EC2: CreateTrafficMirrorTarget
	EC2: CreateTransitGateway
	EC2: CreateTransitGatewayConnect
	EC2: CreateTransitGatewayConnectPeer
	EC2: CreateTransitGatewayMeteringPolicy
	EC2: CreateTransitGatewayMeteringPolicyEntry
	EC2: CreateTransitGatewayMulticastDomain
	EC2: CreateTransitGatewayPeeringAttachment
	EC2: CreateTransitGatewayPolicyTable
	EC2: CreateTransitGatewayPrefixListReference
	EC2: CreateTransitGatewayRoute

Awalan layanan	Tindakan
	EC2: CreateTransitGatewayRouteTable
	EC2: CreateTransitGatewayRouteTableAnnouncement
	EC2: CreateTransitGatewayVpcAttachment
	EC2: CreateVerifiedAccessEndpoint
	EC2: CreateVerifiedAccessGroup
	EC2: CreateVerifiedAccessInstance
	EC2: CreateVerifiedAccessTrustProvider
	EC2: CreateVolume
	EC2: CreateVpc
	EC2: CreateVpcBlockPublicAccessExclusion
	EC2: CreateVpcEncryptionControl
	EC2: CreateVpcEndpoint
	EC2: CreateVpcEndpointConnectionNotification
	EC2: CreateVpcEndpointServiceConfiguration
	EC2: CreateVpcPeeringConnection
	EC2: CreateVpnConcentrator
	EC2: CreateVpnConnection
	EC2: CreateVpnConnectionRoute
	EC2: CreateVpnGateway
	EC2: DeleteCapacityManagerDataExport
	EC2: DeleteCarrierGateway

Awalan layanan	Tindakan
	EC2: DeleteClientVpnEndpoint
	EC2: DeleteClientVpnRoute
	EC2: DeleteCoipCidr
	EC2: DeleteCoipPool
	EC2: DeleteCustomerGateway
	EC2: DeleteDhcpOptions
	EC2: DeleteEgressOnlyInternetGateway
	EC2: DeleteFleets
	EC2: DeleteFlowLogs
	EC2: DeleteFpgaImage
	EC2: DeleteImageUsageReport
	EC2: DeleteInstanceConnectEndpoint
	EC2: DeleteInstanceEventWindow
	EC2: DeleteInternetGateway
	EC2: DeleteIpam
	EC2: DeleteIpamExternalResourceVerificationToken
	EC2: DeleteIpamPolicy
	EC2: DeleteIpamPool
	EC2: DeleteIpamPrefixListResolver
	EC2: DeleteIpamPrefixListResolverTarget
	EC2: DeleteIpamResourceDiscovery

Awalan layanan	Tindakan
	EC2: DeleteIpamScope
	EC2: DeleteKeyPair
	EC2: DeleteLaunchTemplate
	EC2: DeleteLaunchTemplateVersions
	EC2: DeleteLocalGatewayRoute
	EC2: DeleteLocalGatewayRouteTable
	EC2: DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	EC2: DeleteLocalGatewayRouteTableVpcAssociation
	EC2: DeleteLocalGatewayVirtualInterface
	EC2: DeleteLocalGatewayVirtualInterfaceGroup
	EC2: DeleteManagedPrefixList
	EC2: DeleteNatGateway
	EC2: DeleteNetworkAcl
	EC2: DeleteNetworkAclEntry
	EC2: DeleteNetworkInsightsAccessScope
	EC2: DeleteNetworkInsightsAccessScopeAnalysis
	EC2: DeleteNetworkInsightsAnalysis
	EC2: DeleteNetworkInsightsPath
	EC2: DeleteNetworkInterface
	EC2: DeleteNetworkInterfacePermission

Awalan layanan	Tindakan
	EC2: DeletePlacementGroup
	EC2: DeletePublicIpv4Pool
	EC2: DeleteQueuedReservedInstances
	EC2: DeleteRoute
	EC2: DeleteRouteServer
	EC2: DeleteRouteServerEndpoint
	EC2: DeleteRouteServerPeer
	EC2: DeleteRouteTable
	EC2: DeleteSecondaryNetwork
	EC2: DeleteSecondarySubnet
	EC2: DeleteSecurityGroup
	EC2: DeleteSpotDatafeedSubscription
	EC2: DeleteSubnet
	EC2: DeleteSubnetCidrReservation
	EC2: DeleteTrafficMirrorFilter
	EC2: DeleteTrafficMirrorFilterRule
	EC2: DeleteTrafficMirrorSession
	EC2: DeleteTrafficMirrorTarget
	EC2: DeleteTransitGateway
	EC2: DeleteTransitGatewayClientVpnAttachment
	EC2: DeleteTransitGatewayConnect

Awalan layanan	Tindakan
	EC2: DeleteTransitGatewayConnectPeer
	EC2: DeleteTransitGatewayMeteringPolicy
	EC2: DeleteTransitGatewayMeteringPolicyEntry
	EC2: DeleteTransitGatewayMulticastDomain
	EC2: DeleteTransitGatewayPeeringAttachment
	EC2: DeleteTransitGatewayPolicyTable
	EC2: DeleteTransitGatewayPrefixListReference
	EC2: DeleteTransitGatewayRoute
	EC2: DeleteTransitGatewayRouteTable
	EC2: DeleteTransitGatewayRouteTableAnnouncement
	EC2: DeleteTransitGatewayVpcAttachment
	EC2: DeleteVerifiedAccessEndpoint
	EC2: DeleteVerifiedAccessGroup
	EC2: DeleteVerifiedAccessInstance
	EC2: DeleteVerifiedAccessTrustProvider
	EC2: DeleteVolume
	EC2: DeleteVpc
	EC2: DeleteVpcBlockPublicAccessExclusion
	EC2: DeleteVpcEncryptionControl
	EC2: DeleteVpcEndpointConnectionNotifications
	EC2: DeleteVpcEndpointServiceConfigurations

Awalan layanan	Tindakan
	EC2: DeleteVpcEndpoints
	EC2: DeleteVpcPeeringConnection
	EC2: DeleteVpnConcentrator
	EC2: DeleteVpnConnection
	EC2: DeleteVpnConnectionRoute
	EC2: DeleteVpnGateway
	EC2: DeprovisionByoipCidr
	EC2: DeprovisionIpamByoasn
	EC2: DeprovisionIpamPoolCidr
	EC2: DeprovisionPublicIpv4PoolCidr
	EC2: DeregisterImage
	EC2: DeregisterInstanceEventNotificationAttributes
	EC2: DeregisterTransitGatewayMulticastGroupMembers
	EC2: DeregisterTransitGatewayMulticastGroupSources
	EC2: DescribeAccountAttributes
	EC2: DescribeAddressTransfers
	EC2: DescribeAddresses
	EC2: DescribeAddressesAttribute
	EC2: DescribeAggregateIdFormat
	EC2: DescribeAvailabilityZones
	EC2: DescribeAwsNetworkPerformanceMetricSubscriptions

Awalan layanan	Tindakan
	EC2: DescribeBundleTasks
	EC2: DescribeByoipCidrs
	EC2: DescribeCapacityBlockExtensionHistory
	EC2: DescribeCapacityBlockExtensionOfferings
	EC2: DescribeCapacityBlockStatus
	EC2: DescribeCapacityBlocks
	EC2: DescribeCapacityManagerDataExports
	EC2: DescribeCapacityReservationBillingRequests
	EC2: DescribeCapacityReservationCancellationQuotes
	EC2: DescribeCapacityReservationFleets
	EC2: DescribeCapacityReservationTopology
	EC2: DescribeCapacityReservations
	EC2: DescribeCarrierGateways
	EC2: DescribeClassicLinkInstances
	EC2: DescribeClientVpnAuthorizationRules
	EC2: DescribeClientVpnConnections
	EC2: DescribeClientVpnEndpoints
	EC2: DescribeClientVpnRoutes
	EC2: DescribeClientVpnTargetNetworks
	EC2: DescribeCoipPools
	EC2: DescribeConversionTasks

Awalan layanan	Tindakan
	EC2: DescribeCustomerGateways
	EC2: DescribeDeclarativePoliciesReports
	EC2: DescribeDhcpOptions
	EC2: DescribeEgressOnlyInternetGateways
	EC2: DescribeElasticGpus
	EC2: DescribeExportImageTasks
	EC2: DescribeExportTasks
	EC2: DescribeFastLaunchImages
	EC2: DescribeFastSnapshotRestores
	EC2: DescribeFleetHistory
	EC2: DescribeFleetInstances
	EC2: DescribeFleets
	EC2: DescribeFlowLogs
	EC2: DescribeFpgaImageAttribute
	EC2: DescribeFpgaImages
	EC2: DescribeHostReservationOfferings
	EC2: DescribeHostReservations
	EC2: DescribeHosts
	EC2: DescribeIamInstanceProfileAssociations
	EC2: DescribeIdFormat
	EC2: DescribeIdentityIdFormat

Awalan layanan	Tindakan
	EC2: DescribeImageAttribute
	EC2: DescribeImageReferences
	EC2: DescribeImageUsageReportEntries
	EC2: DescribeImageUsageReports
	EC2: DescribeImportImageTasks
	EC2: DescribeImportSnapshotTasks
	EC2: DescribeInstanceConnectEndpoints
	EC2: DescribeInstanceCreditSpecifications
	EC2: DescribeInstanceEventNotificationAttributes
	EC2: DescribeInstanceEventWindows
	EC2: DescribeInstanceImageMetadata
	EC2: DescribeInstanceSqlHaHistoryStates
	EC2: DescribeInstanceSqlHaStates
	EC2: DescribeInstanceTopology
	EC2: DescribeInstanceTypes
	EC2: DescribeInternetGateways
	EC2: DescribeIpamByoasn
	EC2: DescribeIpamExternalResourceVerificationTokens
	EC2: DescribeIpamPolicies
	EC2: DescribeIpamPoolAllocations
	EC2: DescribeIpamPools

Awalan layanan	Tindakan
	EC2: DescribeIpamPrefixListResolverTargets
	EC2: DescribeIpamPrefixListResolvers
	EC2: DescribeIpamResourceDiscoveries
	EC2: DescribeIpamResourceDiscoveryAssociations
	EC2: DescribeIpamScopes
	EC2: DescribeIpams
	EC2: DescribeIpv6Pools
	EC2: DescribeKeyPairs
	EC2: DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	EC2: DescribeLocalGatewayRouteTableVpcAssociations
	EC2: DescribeLocalGatewayRouteTables
	EC2: DescribeLocalGatewayVirtualInterfaceGroups
	EC2: DescribeLocalGatewayVirtualInterfaces
	EC2: DescribeLocalGateways
	EC2: DescribeLockedSnapshots
	EC2: DescribeMacHosts
	EC2: DescribeMacModificationTasks
	EC2: DescribeManagedPrefixLists
	EC2: DescribeMovingAddresses
	EC2: DescribeNatGateways

Awalan layanan	Tindakan
	EC2: DescribeNetworkAcls
	EC2: DescribeNetworkInsightsAccessScopeAnalyses
	EC2: DescribeNetworkInsightsAccessScopes
	EC2: DescribeNetworkInsightsAnalyses
	EC2: DescribeNetworkInsightsPaths
	EC2: DescribeNetworkInterfaceAttribute
	EC2: DescribeNetworkInterfacePermissions
	EC2: DescribeNetworkInterfaces
	EC2: DescribeOutpostLags
	EC2: DescribePlacementGroups
	EC2: DescribePrefixLists
	EC2: DescribePrincipalIdFormat
	EC2: DescribePublicIpv4Pools
	EC2: DescribeRegions
	EC2: DescribeReplaceRootVolumeTasks
	EC2: DescribeReservedInstances
	EC2: DescribeReservedInstancesListings
	EC2: DescribeReservedInstancesModifications
	EC2: DescribeReservedInstancesOfferings
	EC2: DescribeRouteServerEndpoints
	EC2: DescribeRouteServerPeers

Awalan layanan	Tindakan
	EC2: DescribeRouteServers
	EC2: DescribeRouteTables
	EC2: DescribeScheduledInstanceAvailability
	EC2: DescribeScheduledInstances
	EC2: DescribeSecondaryInterfaces
	EC2: DescribeSecondaryNetworks
	EC2: DescribeSecondarySubnets
	EC2: DescribeSecurityGroupReferences
	EC2: DescribeSecurityGroupRules
	EC2: DescribeSecurityGroupVpcAssociations
	EC2: DescribeSecurityGroups
	EC2: DescribeServiceLinkVirtualInterfaces
	EC2: DescribeSnapshotAttribute
	EC2: DescribeSnapshotTierStatus
	EC2: DescribeSpotDatafeedSubscription
	EC2: DescribeSpotFleetInstances
	EC2: DescribeSpotFleetRequestHistory
	EC2: DescribeSpotFleetRequests
	EC2: DescribeSpotInstanceRequests
	EC2: DescribeSpotPriceHistory
	EC2: DescribeStaleSecurityGroups

Awalan layanan	Tindakan
	EC2: DescribeStoreImageTasks
	EC2: DescribeTrafficMirrorFilterRules
	EC2: DescribeTrafficMirrorFilters
	EC2: DescribeTrafficMirrorSessions
	EC2: DescribeTrafficMirrorTargets
	EC2: DescribeTransitGatewayAttachments
	EC2: DescribeTransitGatewayConnectPeers
	EC2: DescribeTransitGatewayConnects
	EC2: DescribeTransitGatewayMeteringPolicies
	EC2: DescribeTransitGatewayMulticastDomains
	EC2: DescribeTransitGatewayPeeringAttachments
	EC2: DescribeTransitGatewayPolicyTables
	EC2: DescribeTransitGatewayRouteTableAnnouncements
	EC2: DescribeTransitGatewayRouteTables
	EC2: DescribeTransitGatewayVpcAttachments
	EC2: DescribeTransitGateways
	EC2: DescribeTrunkInterfaceAssociations
	EC2: DescribeVerifiedAccessEndpoints
	EC2: DescribeVerifiedAccessGroups
	EC2: DescribeVerifiedAccessInstanceLoggingConfigurations
	EC2: DescribeVerifiedAccessInstances

Awalan layanan	Tindakan
	EC2: DescribeVerifiedAccessTrustProviders
	EC2: DescribeVolumeAttribute
	EC2: DescribeVolumeStatus
	EC2: DescribeVolumes
	EC2: DescribeVolumesModifications
	EC2: DescribeVpcAttribute
	EC2: DescribeVpcBlockPublicAccessExclusions
	EC2: DescribeVpcBlockPublicAccessOptions
	EC2: DescribeVpcClassicLink
	EC2: DescribeVpcClassicLinkDnsSupport
	EC2: DescribeVpcEncryptionControls
	EC2: DescribeVpcEndpointAssociations
	EC2: DescribeVpcEndpointConnectionNotifications
	EC2: DescribeVpcEndpointConnections
	EC2: DescribeVpcEndpointServiceConfigurations
	EC2: DescribeVpcEndpointServicePermissions
	EC2: DescribeVpcEndpointServices
	EC2: DescribeVpcEndpoints
	EC2: DescribeVpcPeeringConnections
	EC2: DescribeVpcs
	EC2: DescribeVpnConcentrators

Awalan layanan	Tindakan
	EC2: DescribeVpnConnections
	EC2: DescribeVpnGateways
	EC2: DetachClassicLinkVpc
	EC2: DetachImageWatermark
	EC2: DetachInternetGateway
	EC2: DetachNetworkInterface
	EC2: DetachVerifiedAccessTrustProvider
	EC2: DetachVolume
	EC2: DetachVpnGateway
	EC2: DisableAddressTransfer
	EC2: DisableAllowedImagesSettings
	EC2: DisableAwsNetworkPerformanceMetricSubscription
	EC2: DisableCapacityManager
	EC2: DisableEbsEncryptionByDefault
	EC2: DisableFastLaunch
	EC2: DisableFastSnapshotRestores
	EC2: DisableImage
	EC2: DisableImageBlockPublicAccess
	EC2: DisableImageDeprecation
	EC2: DisableImageDeregistrationProtection
	EC2: DisableInstanceSqlHaStandbyDetections

Awalan layanan	Tindakan
	EC2: DisableIpamOrganizationAdminAccount
	EC2: DisableIpamPolicy
	EC2: DisableRouteServerPropagation
	EC2: DisableSerialConsoleAccess
	EC2: DisableSnapshotBlockPublicAccess
	EC2: DisableTransitGatewayRouteTablePropagation
	EC2: DisableVgwRoutePropagation
	EC2: DisableVpcClassicLink
	EC2: DisableVpcClassicLinkDnsSupport
	EC2: DisassociateAddress
	EC2: DisassociateCapacityReservationBillingOwner
	EC2: DisassociateClientVpnTargetNetwork
	EC2: DisassociateEnclaveCertificateIamRole
	EC2: DisassociateIamInstanceProfile
	EC2: DisassociateInstanceEventWindow
	EC2: DisassociateIpamByoasn
	EC2: DisassociateIpamResourceDiscovery
	EC2: DisassociateNatGatewayAddress
	EC2: DisassociateRouteServer
	EC2: DisassociateRouteTable
	EC2: DisassociateSecurityGroupVpc

Awalan layanan	Tindakan
	EC2: DisassociateSubnetCidrBlock
	EC2: DisassociateTransitGatewayMulticastDomain
	EC2: DisassociateTransitGatewayPolicyTable
	EC2: DisassociateTransitGatewayRouteTable
	EC2: DisassociateTrunkInterface
	EC2: DisassociateVpcCidrBlock
	EC2: EnableAddressTransfer
	EC2: EnableAllowedImagesSettings
	EC2: EnableAwsNetworkPerformanceMetricSubscription
	EC2: EnableCapacityManager
	EC2: EnableEbsEncryptionByDefault
	EC2: EnableFastLaunch
	EC2: EnableFastSnapshotRestores
	EC2: EnableImage
	EC2: EnableImageBlockPublicAccess
	EC2: EnableImageDeprecation
	EC2: EnableImageDeregistrationProtection
	EC2: EnableInstanceSqlHaStandbyDetections
	EC2: EnableIpamOrganizationAdminAccount
	EC2: EnableIpamPolicy
	EC2: EnableReachabilityAnalyzerOrganizationSharing

Awalan layanan	Tindakan
	EC2: EnableRouteServerPropagation
	EC2: EnableSerialConsoleAccess
	EC2: EnableSnapshotBlockPublicAccess
	EC2: EnableTransitGatewayRouteTablePropagation
	EC2: EnableVgwRoutePropagation
	ec2: EnableVolume IO
	EC2: EnableVpcClassicLink
	EC2: EnableVpcClassicLinkDnsSupport
	EC2: ExportClientVpnClientCertificateRevocationList
	EC2: ExportClientVpnClientConfiguration
	EC2: ExportImage
	EC2: ExportTransitGatewayRoutes
	EC2: ExportVerifiedAccessInstanceClientConfiguration
	EC2: GetActiveVpnTunnelStatus
	EC2: GetAllowedImagesSettings
	EC2: GetAssociatedEnclaveCertificateIamRoles
	EC2: GetAssociatedIpv6PoolCidrs
	EC2: GetAwsNetworkPerformanceData
	EC2: GetCapacityManagerAttributes
	EC2: GetCapacityManagerMetricData
	EC2: GetCapacityManagerMetricDimensions

Awalan layanan	Tindakan
	EC2: GetCapacityReservationUsage
	EC2: GetCoipPoolUsage
	EC2: GetConsoleOutput
	EC2: GetConsoleScreenshot
	EC2: GetDeclarativePoliciesReportSummary
	EC2: GetDefaultCreditSpecification
	EC2: GetEbsDefaultKmsKeyId
	EC2: GetEbsEncryptionByDefault
	EC2: GetEnabledIpamPolicy
	EC2: GetFlowLogsIntegrationTemplate
	EC2: GetGroupsForCapacityReservation
	EC2: GetHostReservationPurchasePreview
	EC2: GetImageAncestry
	EC2: GetImageBlockPublicAccessState
	EC2: GetInstanceMetadataDefaults
	EC2: GetInstanceTpmEkPub
	EC2: GetInstanceTypesFromInstanceRequirements
	EC2: GetInstanceUefiData
	EC2: GetIpamAddressHistory
	EC2: GetIpamDiscoveredAccounts
	EC2: GetIpamDiscoveredPublicAddresses

Awalan layanan	Tindakan
	EC2: GetIpamDiscoveredResourceCidrs
	EC2: GetIpamPolicyAllocationRules
	EC2: GetIpamPolicyOrganizationTargets
	EC2: GetIpamPoolAllocations
	EC2: GetIpamPoolCidrs
	EC2: GetIpamPrefixListResolverRules
	EC2: GetIpamPrefixListResolverVersionEntries
	EC2: GetIpamPrefixListResolverVersions
	EC2: GetIpamResourceCidrs
	EC2: GetLaunchTemplateData
	EC2: GetManagedPrefixListAssociations
	EC2: GetManagedPrefixListEntries
	EC2: GetManagedResourceVisibility
	EC2: GetNetworkInsightsAccessScopeAnalysisFindings
	EC2: GetNetworkInsightsAccessScopeContent
	EC2: GetPasswordData
	EC2: GetReservedInstancesExchangeQuote
	EC2: GetRouteServerAssociations
	EC2: GetRouteServerPropagations
	EC2: GetRouteServerRoutingDatabase
	EC2: GetSecurityGroupsForVpc

Awalan layanan	Tindakan
	EC2: GetSerialConsoleAccessStatus
	EC2: GetSnapshotBlockPublicAccessState
	EC2: GetSpotPlacementScores
	EC2: GetSubnetCidrReservations
	EC2: GetTransitGatewayAttachmentPropagations
	EC2: GetTransitGatewayMeteringPolicyEntries
	EC2: GetTransitGatewayMulticastDomainAssociations
	EC2: GetTransitGatewayPolicyTableAssociations
	EC2: GetTransitGatewayPolicyTableEntries
	EC2: GetTransitGatewayPrefixListReferences
	EC2: GetTransitGatewayRouteTableAssociations
	EC2: GetTransitGatewayRouteTablePropagations
	EC2: GetVerifiedAccessEndpointPolicy
	EC2: GetVerifiedAccessEndpointTargets
	EC2: GetVerifiedAccessGroupPolicy
	EC2: GetVpcResourcesBlockingEncryptionEnforcement
	EC2: GetVpnConnectionDeviceSampleConfiguration
	EC2: GetVpnConnectionDeviceTypes
	EC2: GetVpnTunnelReplacementStatus
	EC2: ImportClientVpnClientCertificateRevocationList
	EC2: ImportImage

Awalan layanan	Tindakan
	EC2: ImportInstance
	EC2: ImportKeyPair
	EC2: ImportSnapshot
	EC2: ImportVolume
	EC2: ListImagesInRecycleBin
	EC2: ListSnapshotsInRecycleBin
	EC2: ListVolumesInRecycleBin
	EC2: LockSnapshot
	EC2: ModifyAddressAttribute
	EC2: ModifyAvailabilityZoneGroup
	EC2: ModifyCapacityReservation
	EC2: ModifyCapacityReservationFleet
	EC2: ModifyClientVpnEndpoint
	EC2: ModifyDefaultCreditSpecification
	EC2: ModifyEbsDefaultKmsKeyId
	EC2: ModifyFleet
	EC2: ModifyFpgaImageAttribute
	EC2: ModifyHosts
	EC2: ModifyIdFormat
	EC2: ModifyIdentityIdFormat
	EC2: ModifyImageAttribute

Awalan layanan	Tindakan
	EC2: ModifyInstanceAttribute
	EC2: ModifyInstanceCapacityReservationAttributes
	EC2: ModifyInstanceConnectEndpoint
	EC2: ModifyInstanceCpuOptions
	EC2: ModifyInstanceCreditSpecification
	EC2: ModifyInstanceEventStartTime
	EC2: ModifyInstanceEventWindow
	EC2: ModifyInstanceMaintenanceOptions
	EC2: ModifyInstanceMetadataDefaults
	EC2: ModifyInstanceMetadataOptions
	EC2: ModifyInstanceNetworkPerformanceOptions
	EC2: ModifyInstancePlacement
	EC2: ModifyIpam
	EC2: ModifyIpamPolicyAllocationRules
	EC2: ModifyIpamPool
	EC2: ModifyIpamPoolAllocation
	EC2: ModifyIpamPrefixListResolver
	EC2: ModifyIpamPrefixListResolverTarget
	EC2: ModifyIpamResourceCidr
	EC2: ModifyIpamResourceDiscovery
	EC2: ModifyIpamScope

Awalan layanan	Tindakan
	EC2: ModifyLaunchTemplate
	EC2: ModifyLocalGatewayRoute
	EC2: ModifyManagedPrefixList
	EC2: ModifyManagedResourceVisibility
	EC2: ModifyNetworkInterfaceAttribute
	EC2: ModifyPrivateDnsNameOptions
	EC2: ModifyPublicIpDnsNameOptions
	EC2: ModifyReservedInstances
	EC2: ModifyRouteServer
	EC2: ModifySecurityGroupRules
	EC2: ModifySnapshotAttribute
	EC2: ModifySnapshotTier
	EC2: ModifySpotFleetRequest
	EC2: ModifySubnetAttribute
	EC2: ModifyTrafficMirrorFilterNetworkServices
	EC2: ModifyTrafficMirrorFilterRule
	EC2: ModifyTrafficMirrorSession
	EC2: ModifyTransitGateway
	EC2: ModifyTransitGatewayMeteringPolicy
	EC2: ModifyTransitGatewayPrefixListReference
	EC2: ModifyTransitGatewayVpcAttachment

Awalan layanan	Tindakan
	EC2: ModifyVerifiedAccessEndpoint
	EC2: ModifyVerifiedAccessEndpointPolicy
	EC2: ModifyVerifiedAccessGroup
	EC2: ModifyVerifiedAccessGroupPolicy
	EC2: ModifyVerifiedAccessInstance
	EC2: ModifyVerifiedAccessInstanceLoggingConfiguration
	EC2: ModifyVerifiedAccessTrustProvider
	EC2: ModifyVolume
	EC2: ModifyVolumeAttribute
	EC2: ModifyVpcAttribute
	EC2: ModifyVpcBlockPublicAccessExclusion
	EC2: ModifyVpcBlockPublicAccessOptions
	EC2: ModifyVpcEncryptionControl
	EC2: ModifyVpcEndpoint
	EC2: ModifyVpcEndpointConnectionNotification
	EC2: ModifyVpcEndpointServiceConfiguration
	EC2: ModifyVpcEndpointServicePayerResponsibility
	EC2: ModifyVpcEndpointServicePermissions
	EC2: ModifyVpcPeeringConnectionOptions
	EC2: ModifyVpcTenancy
	EC2: ModifyVpnConnection

Awalan layanan	Tindakan
	EC2: ModifyVpnConnectionOptions
	EC2: ModifyVpnTunnelCertificate
	EC2: ModifyVpnTunnelOptions
	EC2: MonitorInstances
	EC2: MoveAddressToVpc
	EC2: MoveByoipCidrToIpam
	EC2: MoveCapacityReservationInstances
	EC2: ProvisionByoipCidr
	EC2: ProvisionIpamByoasn
	EC2: ProvisionIpamPoolCidr
	EC2: ProvisionPublicIpv4PoolCidr
	EC2: PurchaseCapacityBlockExtension
	EC2: PurchaseHostReservation
	EC2: PurchaseReservedInstancesOffering
	EC2: PurchaseScheduledInstances
	EC2: RebootInstances
	EC2: RegisterImage
	EC2: RegisterInstanceEventNotificationAttributes
	EC2: RegisterTransitGatewayMulticastGroupMembers
	EC2: RegisterTransitGatewayMulticastGroupSources
	EC2: RejectCapacityReservationBillingOwnership

Awalan layanan	Tindakan
	EC2: RejectTransitGatewayClientVpnAttachment
	EC2: RejectTransitGatewayMulticastDomainAssociations
	EC2: RejectTransitGatewayPeeringAttachment
	EC2: RejectTransitGatewayVpcAttachment
	EC2: RejectVpcEndpointConnections
	EC2: RejectVpcPeeringConnection
	EC2: ReleaseAddress
	EC2: ReleaseHosts
	EC2: ReleaseIpamPoolAllocation
	EC2: ReplaceIamInstanceProfileAssociation
	EC2: ReplaceImageCriteriaInAllowedImagesSettings
	EC2: ReplaceNetworkAclAssociation
	EC2: ReplaceNetworkAclEntry
	EC2: ReplaceRoute
	EC2: ReplaceRouteTableAssociation
	EC2: ReplaceTransitGatewayRoute
	EC2: ReplaceVpnTunnel
	EC2: ReportInstanceStatus
	EC2: RequestSpotFleet
	EC2: RequestSpotInstances
	EC2: ResetAddressAttribute

Awalan layanan	Tindakan
	EC2: ResetEbsDefaultKmsKeyId
	EC2: ResetFpgaImageAttribute
	EC2: ResetImageAttribute
	EC2: ResetInstanceAttribute
	EC2: ResetNetworkInterfaceAttribute
	EC2: ResetSnapshotAttribute
	EC2: RestoreAddressToClassic
	EC2: RestoreImageFromRecycleBin
	EC2: RestoreManagedPrefixListVersion
	EC2: RestoreSnapshotFromRecycleBin
	EC2: RestoreSnapshotTier
	EC2: RestoreVolumeFromRecycleBin
	EC2: RevokeClientVpnIngress
	EC2: RevokeSecurityGroupEgress
	EC2: RevokeSecurityGroupIngress
	EC2: RunInstances
	EC2: RunScheduledInstances
	EC2: SearchLocalGatewayRoutes
	EC2: SearchTransitGatewayMulticastGroups
	EC2: SearchTransitGatewayRoutes
	EC2: SendDiagnosticInterrupt

Awalan layanan	Tindakan
	EC2: StartDeclarativePoliciesReport
	EC2: StartInstances
	EC2: StartNetworkInsightsAccessScopeAnalysis
	EC2: StartNetworkInsightsAnalysis
	EC2: StartVpcEndpointServicePrivateDnsVerification
	EC2: TerminateClientVpnConnections
	EC2: UnassignIpv6Addresses
	EC2: UnassignPrivateIpAddresses
	EC2: UnassignPrivateNatGatewayAddress
	EC2: UnlockSnapshot
	EC2: UnmonitorInstances
	EC2: UpdateCapacityManagerOrganizationsAccess
	EC2: UpdateInterruptibleCapacityReservationAllocation
	EC2: UpdateSecurityGroupRuleDescriptionsEgress
	EC2: UpdateSecurityGroupRuleDescriptionsIngress
	EC2: WithdrawByoipCidr

Awalan layanan	Tindakan
ecr	ecr: BatchCheckLayerAvailability ecr: BatchDeleteImage ecr: BatchGetImage ecr: BatchGetRepositoryScanningConfiguration ecr: CompleteLayerUpload ecr: CreatePullThroughCacheRule ecr: CreateRepositoryCreationTemplate ecr: DeleteLifecyclePolicy ecr: DeletePullThroughCacheRule ecr: DeleteRegistryPolicy ecr: DeleteRepository ecr: DeleteRepositoryCreationTemplate ecr: DeleteRepositoryPolicy ecr: DeleteSigningConfiguration ecr: DescribeImageReplicationStatus ecr: DescribeImageScanFindings ecr: DescribeImages ecr: DescribePullThroughCacheRules ecr: DescribeRegistry ecr: DescribeRepositories ecr: DescribeRepositoryCreationTemplates

Awalan layanan	Tindakan
	ecr: GetAccountSetting ecr: GetAuthorizationToken ecr: GetDownloadUrlForLayer ecr: GetLifecyclePolicy ecr: GetLifecyclePolicyPreview ecr: GetRegistryPolicy ecr: GetRegistryScanningConfiguration ecr: GetRepositoryPolicy ecr: GetSigningConfiguration ecr: InitiateLayerUpload ecr: ListImages ecr: ListPullTimeUpdateExclusions ecr: PutAccountSetting ecr: PutImage ecr: PutImageScanningConfiguration ecr: PutRegistryPolicy ecr: PutRegistryScanningConfiguration ecr: PutReplicationConfiguration ecr: StartImageScan ecr: StartLifecyclePolicyPreview ecr: UpdatePullThroughCacheRule

Awalan layanan	Tindakan
	ecr: UpdateRepositoryCreationTemplate ecr: UploadLayerPart ecr: ValidatePullThroughCacheRule

Awalan layanan	Tindakan
ecr-publik	ecr-publik: BatchCheckLayerAvailability
	ecr-publik: BatchDeleteImage
	ecr-publik: CompleteLayerUpload
	ecr-publik: CreateRepository
	ecr-publik: DeleteRepository
	ecr-publik: DeleteRepositoryPolicy
	ecr-publik: DescribeImages
	ecr-publik: DescribeRegistries
	ecr-publik: DescribeRepositories
	ecr-publik: GetAuthorizationToken
	ecr-publik: GetRegistryCatalogData
	ecr-publik: GetRepositoryCatalogData
	ecr-publik: GetRepositoryPolicy
	ecr-publik: InitiateLayerUpload
	ecr-publik: PutImage
	ecr-publik: PutRegistryCatalogData
	ecr-publik: PutRepositoryCatalogData
	ecr-publik: SetRepositoryPolicy
	ecr-publik: UploadLayerPart

Awalan layanan	Tindakan
ecs	ecs: ContinueServiceDeployment ecs: CreateCapacityProvider ecs: CreateCluster ecs: CreateDaemon ecs: CreateService ecs: CreateTaskSet ecs: DeleteAccountSetting ecs: DeleteAttributes ecs: DeleteCapacityProvider ecs: DeleteCluster ecs: DeleteDaemon ecs: DeleteDaemonTaskDefinition ecs: DeleteExpressGatewayService ecs: DeleteService ecs: DeleteTaskDefinitions ecs: DeleteTaskSet ecs: DeregisterContainerInstance ecs: DeregisterTaskDefinition ecs: DescribeCapacityProviders ecs: DescribeClusters ecs: DescribeContainerInstances

Awalan layanan	Tindakan
	ecs: DescribeDaemon ecs: DescribeDaemonDeployments ecs: DescribeDaemonRevisions ecs: DescribeDaemonTaskDefinition ecs: DescribeExpressGatewayService ecs: DescribeServiceDeployments ecs: DescribeServiceRevisions ecs: DescribeServices ecs: DescribeTaskDefinition ecs: DescribeTaskSets ecs: DescribeTasks ecs: DiscoverPollEndpoint ecs: ExecuteCommand ecs: GetTaskProtection ecs: ListAccountSettings ecs: ListAttributes ecs: ListClusters ecs: ListContainerInstances ecs: ListDaemonDeployments ecs: ListDaemonTaskDefinitions ecs: ListDaemons

Awalan layanan	Tindakan
	ecs: ListServiceDeployments ecs: ListServices ecs: ListServicesByNamespace ecs: ListTaskDefinitionFamilies ecs: ListTaskDefinitions ecs: ListTasks ecs: PutAccountSetting ecs: PutAccountSettingDefault ecs: PutAttributes ecs: PutClusterCapacityProviders ecs: RegisterContainerInstance ecs: RegisterDaemonTaskDefinition ecs: RunTask ecs: StartTask ecs: StopServiceDeployment ecs: StopTask ecs: SubmitAttachmentStateChanges ecs: SubmitContainerStateChange ecs: SubmitTaskStateChange ecs: UpdateCapacityProvider ecs: UpdateCluster

Awalan layanan	Tindakan
	ecs: UpdateClusterSettings ecs: UpdateContainerAgent ecs: UpdateContainerInstancesState ecs: UpdateDaemon ecs: UpdateExpressGatewayService ecs: UpdateService ecs: UpdateServicePrimaryTaskSet ecs: UpdateTaskProtection ecs: UpdateTaskSet

Awalan layanan	Tindakan
eks	eks: AssociateAccessPolicy
	eks: AssociateEncryptionConfig
	eks: AssociateIdentityProviderConfig
	eks: CancelUpdate
	eks: CreateAccessEntry
	eks: CreateAddon
	eks: CreateCluster
	eks: CreateEksAnywhereSubscription
	eks: CreateFargateProfile
	eks: CreateNodegroup
	eks: DeleteAccessEntry
	eks: DeleteAddon
	eks: DeleteCapability
	eks: DeleteCluster
	eks: DeleteEksAnywhereSubscription
	eks: DeleteFargateProfile
	eks: DeleteNodegroup
	eks: DeletePodIdentityAssociation
	eks: DeregisterCluster
	eks: DescribeAccessEntry
	eks: DescribeAddon

Awalan layanan	Tindakan
	eks: DescribeAddonConfiguration
	eks: DescribeAddonVersions
	eks: DescribeCapability
	eks: DescribeCluster
	eks: DescribeClusterVersions
	eks: DescribeEksAnywhereSubscription
	eks: DescribeFargateProfile
	eks: DescribeIdentityProviderConfig
	eks: DescribeInsight
	eks: DescribeInsightsRefresh
	eks: DescribeNodegroup
	eks: DescribePodIdentityAssociation
	eks: DescribeUpdate
	eks: DisassociateAccessPolicy
	eks: DisassociateIdentityProviderConfig
	eks: ListAccessEntries
	eks: ListAccessPolicies
	eks: ListAddons
	eks: ListAssociatedAccessPolicies
	eks: ListCapabilities
	eks: ListClusters

Awalan layanan	Tindakan
	eks: ListEksAnywhereSubscriptions
	eks: ListFargateProfiles
	eks: ListIdentityProviderConfigs
	eks: ListInsights
	eks: ListNodegroups
	eks: ListPodIdentityAssociations
	eks: ListUpdates
	eks: RegisterCluster
	eks: StartInsightsRefresh
	eks: UpdateAccessEntry
	eks: UpdateAddon
	eks: UpdateCapability
	eks: UpdateClusterConfig
	eks: UpdateClusterVersion
	eks: UpdateEksAnywhereSubscription
	eks: UpdateNodegroupConfig
	eks: UpdateNodegroupVersion
	eks: UpdatePodIdentityAssociation

Awalan layanan	Tindakan
elasticache	elastisakit: AuthorizeCacheSecurityGroupIngress elastisakit: BatchApplyUpdateAction elastisakit: BatchStopUpdateAction elastisakit: CompleteMigration elastisakit: CopyServerlessCacheSnapshot elastisakit: CopySnapshot elastisakit: CreateCacheCluster elastisakit: CreateCacheParameterGroup elastisakit: CreateCacheSecurityGroup elastisakit: CreateCacheSubnetGroup elastisakit: CreateGlobalReplicationGroup elastisakit: CreateReplicationGroup elastisakit: CreateServerlessCache elastisakit: CreateServerlessCacheSnapshot elastisakit: CreateSnapshot elastisakit: CreateUser elastisakit: CreateUserGroup elastisakit: DecreaseNodeGroupsInGlobalReplicationGroup elastisakit: DecreaseReplicaCount elastisakit: DeleteCacheCluster elastisakit: DeleteCacheParameterGroup

Awalan layanan	Tindakan
	elastisakit: DeleteCacheSecurityGroup elastisakit: DeleteCacheSubnetGroup elastisakit: DeleteGlobalReplicationGroup elastisakit: DeleteReplicationGroup elastisakit: DeleteServerlessCache elastisakit: DeleteServerlessCacheSnapshot elastisakit: DeleteSnapshot elastisakit: DeleteUser elastisakit: DeleteUserGroup elastisakit: DescribeCacheClusters elastisakit: DescribeCacheEngineVersions elastisakit: DescribeCacheParameterGroups elastisakit: DescribeCacheParameters elastisakit: DescribeCacheSecurityGroups elastisakit: DescribeCacheSubnetGroups elastisakit: DescribeEngineDefaultParameters elastisakit: DescribeEvents elastisakit: DescribeGlobalReplicationGroups elastisakit: DescribeReplicationGroups elastisakit: DescribeReservedCacheNodes elastisakit: DescribeReservedCacheNodesOfferings

Awalan layanan	Tindakan
	elastisakit: DescribeServerlessCacheSnapshots elastisakit: DescribeServerlessCaches elastisakit: DescribeServiceUpdates elastisakit: DescribeSnapshots elastisakit: DescribeUpdateActions elastisakit: DescribeUserGroups elastisakit: DescribeUsers elastisakit: DisassociateGlobalReplicationGroup elastisakit: ExportServerlessCacheSnapshot elastisakit: FailoverGlobalReplicationGroup elastisakit: IncreaseNodeGroupsInGlobalReplicationGroup elastisakit: IncreaseReplicaCount elastisakit: ListAllowedNodeTypeModifications elastisakit: ModifyCacheCluster elastisakit: ModifyCacheParameterGroup elastisakit: ModifyCacheSubnetGroup elastisakit: ModifyGlobalReplicationGroup elastisakit: ModifyReplicationGroup elastisakit: ModifyReplicationGroupShardConfiguration elastisakit: ModifyServerlessCache elastisakit: ModifyUser

Awalan layanan	Tindakan
	elastisakit: ModifyUserGroup elastisakit: PurchaseReservedCacheNodesOffering elastisakit: RebalanceSlotsInGlobalReplicationGroup elastisakit: RebootCacheCluster elastisakit: ResetCacheParameterGroup elastisakit: RevokeCacheSecurityGroupIngress elastisakit: StartMigration elastisakit: TestFailover elastisakit: TestMigration

Awalan layanan	Tindakan
tangkai beanstalk	tangkai elastis: AbortEnvironmentUpdate
	tangkai elastis: ApplyEnvironmentManagedAction
	tangkai elastis: AssociateEnvironmentOperationsRole
	ElasticBeanstalk:CheckDNSAvailability
	tangkai elastis: ComposeEnvironments
	tangkai elastis: CreateApplication
	tangkai elastis: CreateApplicationVersion
	tangkai elastis: CreateConfigurationTemplate
	tangkai elastis: CreateEnvironment
	tangkai elastis: CreatePlatformVersion
	tangkai elastis: CreateStorageLocation
	tangkai elastis: DeleteApplication
	tangkai elastis: DeleteApplicationVersion
	tangkai elastis: DeleteConfigurationTemplate
	tangkai elastis: DeleteEnvironmentConfiguration
	tangkai elastis: DeletePlatformVersion
	tangkai elastis: DescribeAccountAttributes
	tangkai elastis: DescribeApplicationVersions
	tangkai elastis: DescribeApplications
	tangkai elastis: DescribeConfigurationOptions
	tangkai elastis: DescribeConfigurationSettings

Awalan layanan	Tindakan
	tangkai elastis: DescribeEnvironmentHealth
	tangkai elastis: DescribeEnvironmentManagedActionHistory
	tangkai elastis: DescribeEnvironmentManagedActions
	tangkai elastis: DescribeEnvironmentResources
	tangkai elastis: DescribeEnvironments
	tangkai elastis: DescribeEvents
	tangkai elastis: DescribeInstancesHealth
	tangkai elastis: DescribePlatformVersion
	tangkai elastis: DisassociateEnvironmentOperationsRole
	tangkai elastis: ListAvailableSolutionStacks
	tangkai elastis: ListPlatformBranches
	tangkai elastis: ListPlatformVersions
	tangkai elastis: RebuildEnvironment
	tangkai elastis: RequestEnvironmentInfo
	tangkai elastis: RestartAppServer
	tangkai elastis: RetrieveEnvironmentInfo
	tangkai elastis: SwapEnvironmentCNAMEs
	tangkai elastis: TerminateEnvironment
	tangkai elastis: UpdateApplication
	tangkai elastis: UpdateApplicationResourceLifecycle
	tangkai elastis: UpdateApplicationVersion

Awalan layanan	Tindakan
	tangkai elastis: UpdateConfigurationTemplate tangkai elastis: UpdateEnvironment tangkai elastis: ValidateConfigurationSettings

Awalan layanan	Tindakan
sistem file elastis	sistem file elastis: CreateAccessPoint sistem file elastis: CreateFileSystem sistem file elastis: CreateMountTarget sistem file elastis: CreateReplicationConfiguration sistem file elastis: DeleteAccessPoint sistem file elastis: DeleteFileSystem sistem file elastis: DeleteFileSystemPolicy sistem file elastis: DeleteMountTarget sistem file elastis: DeleteReplicationConfiguration sistem file elastis: DescribeAccessPoints sistem file elastis: DescribeAccountPreferences sistem file elastis: DescribeBackupPolicy sistem file elastis: DescribeFileSystemPolicy sistem file elastis: DescribeFileSystems sistem file elastis: DescribeLifecycleConfiguration sistem file elastis: DescribeMountTargetSecurityGroups sistem file elastis: DescribeMountTargets sistem file elastis: DescribeReplicationConfigurations sistem file elastis: ModifyMountTargetSecurityGroups sistem file elastis: PutAccountPreferences sistem file elastis: PutBackupPolicy

Awalan layanan	Tindakan
	sistem file elastis: PutFileSystemPolicy sistem file elastis: PutLifecycleConfiguration sistem file elastis: UpdateFileSystem sistem file elastis: UpdateFileSystemProtection

Awalan layanan	Tindakan
elasticloadbalancing	elasticloadbalancing: AddListenerCertificates
	elasticloadbalancing: AddTrustStoreRevocations
	elasticloadbalancing: ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing: AttachLoadBalancerToSubnets
	elasticloadbalancing: ConfigureHealthCheck
	elasticloadbalancing: CreateAppCookieStickinessPolicy
	elasticloadbalancing: CreateLBCookieStickinessPolicy
	elasticloadbalancing: CreateListener
	elasticloadbalancing: CreateLoadBalancer
	elasticloadbalancing: CreateLoadBalancerListeners
	elasticloadbalancing: CreateLoadBalancerPolicy
	elasticloadbalancing: CreateRule
	elasticloadbalancing: CreateTargetGroup
	elasticloadbalancing: CreateTrustStore
	elasticloadbalancing: DeleteListener
	elasticloadbalancing: DeleteLoadBalancer
	elasticloadbalancing: DeleteLoadBalancerListeners
	elasticloadbalancing: DeleteLoadBalancerPolicy
	elasticloadbalancing: DeleteRule
	elasticloadbalancing: DeleteSharedTrustStoreAssociation
	elasticloadbalancing: DeleteTargetGroup

Awalan layanan	Tindakan
	elasticloadbalancing: DeleteTrustStore
	elasticloadbalancing: DeregisterInstancesFromLoadBalancer
	elasticloadbalancing: DeregisterTargets
	elasticloadbalancing: DescribeAccountLimits
	elasticloadbalancing: DescribeCapacityReservation
	elasticloadbalancing: DescribeInstanceHealth
	elasticloadbalancing: DescribeListenerAttributes
	elasticloadbalancing: DescribeListenerCertificates
	elasticloadbalancing: DescribeListeners
	elasticloadbalancing: DescribeLoadBalancerAttributes
	elasticloadbalancing: DescribeLoadBalancerPolicies
	elasticloadbalancing: DescribeLoadBalancerPolicyTypes
	elasticloadbalancing: DescribeLoadBalancers
	elasticloadbalancing: DescribeRules
	ElasticLoadBalancing:DescribessIPolicies
	elasticloadbalancing: DescribeTargetGroupAttributes
	elasticloadbalancing: DescribeTargetGroups
	elasticloadbalancing: DescribeTargetHealth
	elasticloadbalancing: DescribeTrustStoreAssociations
	elasticloadbalancing: DescribeTrustStoreRevocations
	elasticloadbalancing: DescribeTrustStores

Awalan layanan	Tindakan
	elasticloadbalancing: DetachLoadBalancerFromSubnets
	elasticloadbalancing: DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing: GetResourcePolicy
	elasticloadbalancing: GetTrustStoreCaCertificatesBundle
	elasticloadbalancing: GetTrustStoreRevocationContent
	elasticloadbalancing: ModifyCapacityReservation
	elasticloadbalancing: ModifyIpPools
	elasticloadbalancing: ModifyListener
	elasticloadbalancing: ModifyLoadBalancerAttributes
	elasticloadbalancing: ModifyRule
	elasticloadbalancing: ModifyTargetGroup
	elasticloadbalancing: ModifyTargetGroupAttributes
	elasticloadbalancing: ModifyTrustStore
	elasticloadbalancing: RegisterInstancesWithLoadBalancer
	elasticloadbalancing: RegisterTargets
	elasticloadbalancing: RemoveListenerCertificates
	elasticloadbalancing: RemoveTrustStoreRevocations
	elasticloadbalancing: SetIpAddressType
	elasticloadbalancing: SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing: SetLoadBalancerPoliciesForBackendServer

Awalan layanan	Tindakan
	elasticloadbalancing: SetLoadBalancerPoliciesOfListener
	elasticloadbalancing: SetRulePriorities
	elasticloadbalancing: SetSecurityGroups
	elasticloadbalancing: SetSubnets

Awalan layanan	Tindakan
wadah emr	wadah emr: CancelJobRun wadah emr: CreateJobTemplate wadah emr: CreateManagedEndpoint wadah emr: CreateSecurityConfiguration wadah emr: CreateVirtualCluster wadah emr: DeleteJobTemplate wadah emr: DeleteManagedEndpoint wadah emr: DeleteSecurityConfiguration wadah emr: DeleteVirtualCluster emr-kontainer: DescribeJobRun emr-kontainer: DescribeJobTemplate emr-kontainer: DescribeManagedEndpoint emr-kontainer: DescribeSecurityConfiguration emr-kontainer: DescribeVirtualCluster emr-kontainer: GetManagedEndpointSessionCredentials emr-kontainer: ListJobRuns emr-kontainer: ListJobTemplates emr-kontainer: ListManagedEndpoints emr-kontainer: ListSecurityConfigurations emr-kontainer: ListVirtualClusters emr-kontainer: StartJobRun

Awalan layanan	Tindakan
emr-tanpa server	emr-tanpa server: CancelJobRun
	emr-tanpa server: CreateApplication
	emr-tanpa server: DeleteApplication
	emr-tanpa server: GetApplication
	emr-tanpa server: GetDashboardForJobRun
	emr-tanpa server: GetJobRun
	emr-tanpa server: GetResourceDashboard
	emr-tanpa server: GetSession
	emr-tanpa server: GetSessionEndpoint
	emr-tanpa server: ListApplications
	emr-tanpa server: ListJobRunAttempts
	emr-tanpa server: ListJobRuns
	emr-tanpa server: ListSessions
	emr-tanpa server: StartApplication
	emr-tanpa server: StartJobRun
	emr-tanpa server: StopApplication
	emr-tanpa server: TerminateSession
	emr-tanpa server: UpdateApplication

Awalan layanan	Tindakan
es	es: AcceptInboundConnection es: AcceptInboundCrossClusterSearchConnection es: AssociatePackage es: AuthorizeVpcEndpointAccess es: CancelElasticsearchServiceSoftwareUpdate es: CancelServiceSoftwareUpdate es: CreateDomain es: CreateElasticsearchDomain es: CreateIndex es: CreateOutboundConnection es: CreateOutboundCrossClusterSearchConnection es: CreatePackage es: CreateVpcEndpoint es: DeleteDomain es: DeleteElasticsearchDomain es: DeleteElasticsearchServiceRole es: DeleteInboundConnection es: DeleteInboundCrossClusterSearchConnection es: DeleteIndex es: DeleteOutboundConnection es: DeleteOutboundCrossClusterSearchConnection

Awalan layanan	Tindakan
	es: DeletePackage es: DeleteVpcEndpoint es: DescribeDomain es: DescribeDomainAutoTunes es: DescribeDomainChangeProgress es: DescribeDomainConfig es: DescribeDomainHealth es: DescribeDomainNodes es: DescribeDomains es: DescribeDryRunProgress es: DescribeElasticsearchDomain es: DescribeElasticsearchDomainConfig es: DescribeElasticsearchDomains es: DescribeElasticsearchInstanceTypeLimits es: DescribeInboundConnections es: DescribeInboundCrossClusterSearchConnections es: DescribeInstanceTypeLimits es: DescribeOutboundConnections es: DescribeOutboundCrossClusterSearchConnections es: DescribePackages es: DescribeReservedElasticsearchInstanceOfferings

Awalan layanan	Tindakan
	es: DescribeReservedElasticsearchInstances es: DescribeReservedInstanceOfferings es: DescribeReservedInstances es: DescribeVpcEndpoints es: DissociatePackage es: DissociatePackages es: GetCompatibleElasticsearchVersions es: GetCompatibleVersions es: GetDataSource es: GetDomainMaintenanceStatus es: GetPackageVersionHistory es: GetUpgradeHistory es: GetUpgradeStatus es: ListDataSources es: ListDomainNames es: ListDomainsForPackage es: ListElasticsearchInstanceTypes es: ListElasticsearchVersions es: ListInsights es: ListInstanceTypeDetails es: ListPackagesForDomain

Awalan layanan	Tindakan
	es: ListScheduledActions es: ListVersions es: ListVpcEndpointAccess es: ListVpcEndpoints es: ListVpcEndpointsForDomain es: PurchaseReservedElasticsearchInstanceOffering es: PurchaseReservedInstanceOffering es: RejectInboundConnection es: RejectInboundCrossClusterSearchConnection es: RevokeVpcEndpointAccess es: RollbackServiceSoftwareUpdate es: StartDomainMaintenance es: StartElasticsearchServiceSoftwareUpdate es: StartServiceSoftwareUpdate es: UpdateDataSource es: UpdateDomainConfig es: UpdateElasticsearchDomainConfig es: UpdateIndex es: UpdatePackage es: UpdatePackageScope es: UpdateScheduledAction

Awalan layanan	Tindakan
	es: UpdateVpcEndpoint es: UpgradeDomain es: UpgradeElasticsearchDomain

Awalan layanan	Tindakan
peristiwa	peristiwa: ActivateEventSource
	peristiwa: CancelReplay
	peristiwa: CreateApiDestination
	peristiwa: CreateArchive
	peristiwa: CreateConnection
	peristiwa: CreateEndpoint
	peristiwa: CreateEventBus
	peristiwa: CreatePartnerEventSource
	peristiwa: DeactivateEventSource
	peristiwa: DeauthorizeConnection
	peristiwa: DeleteApiDestination
	peristiwa: DeleteArchive
	peristiwa: DeleteConnection
	peristiwa: DeleteEndpoint
	peristiwa: DeleteEventBus
	peristiwa: DeletePartnerEventSource
	peristiwa: DeleteRule
	peristiwa: DescribeApiDestination
	peristiwa: DescribeArchive
	peristiwa: DescribeConnection
	peristiwa: DescribeEndpoint

Awalan layanan	Tindakan
	peristiwa: DescribeEventBus
	peristiwa: DescribeEventSource
	peristiwa: DescribePartnerEventSource
	peristiwa: DescribeReplay
	peristiwa: DescribeRule
	peristiwa: DisableRule
	peristiwa: EnableRule
	peristiwa: ListApiDestinations
	peristiwa: ListArchives
	peristiwa: ListConnections
	peristiwa: ListEndpoints
	peristiwa: ListEventBuses
	peristiwa: ListEventSources
	peristiwa: ListPartnerEventSourceAccounts
	peristiwa: ListPartnerEventSources
	peristiwa: ListReplays
	peristiwa: ListRuleNamesByTarget
	peristiwa: ListRules
	peristiwa: ListTargetsByRule
	peristiwa: PutPermission
	peristiwa: PutRule

Awalan layanan	Tindakan
	peristiwa: PutTargets
	peristiwa: RemovePermission
	peristiwa: RemoveTargets
	peristiwa: StartReplay
	peristiwa: TestEventPattern
	peristiwa: UpdateApiDestination
	peristiwa: UpdateArchive
	peristiwa: UpdateConnection
	peristiwa: UpdateEndpoint
	peristiwa: UpdateEventBus

Awalan layanan	Tindakan
ruang finspace	ruang finspace: CreateEnvironment ruang finspace: CreateKxChangeset ruang finspace: CreateKxCluster ruang finspace: CreateKxDatabase ruang finspace: CreateKxDataview ruang finspace: CreateKxEnvironment ruang finspace: CreateKxScalingGroup ruang finspace: CreateKxUser ruang finspace: CreateKxVolume ruang finspace: CreateUser ruang finspace: DeleteEnvironment ruang finspace: DeleteKxCluster ruang finspace: DeleteKxClusterNode ruang finspace: DeleteKxDatabase ruang finspace: DeleteKxDataview ruang finspace: DeleteKxEnvironment ruang finspace: DeleteKxScalingGroup ruang finspace: DeleteKxUser ruang finspace: DeleteKxVolume ruang finspace: GetEnvironment ruang finspace: GetKxChangeset

Awalan layanan	Tindakan
	ruang finspace: GetKxCluster ruang finspace: GetKxConnectionString ruang finspace: GetKxDatabase ruang finspace: GetKxDataview ruang finspace: GetKxEnvironment ruang finspace: GetKxScalingGroup ruang finspace: GetKxUser ruang finspace: GetKxVolume ruang finspace: GetLoadSampleDataSetGroupIntoEnvironmentStatus ruang finspace: GetUser ruang finspace: ListEnvironments ruang finspace: ListKxChangesets ruang finspace: ListKxClusterNodes ruang finspace: ListKxClusters ruang finspace: ListKxDatabases ruang finspace: ListKxDataviews ruang finspace: ListKxEnvironments ruang finspace: ListKxScalingGroups ruang finspace: ListKxUsers ruang finspace: ListKxVolumes

Awalan layanan	Tindakan
	ruang finspace: ListUsers ruang finspace: LoadSampleDataSetGroupIntoEnvironment ruang finspace: ResetUserPassword ruang finspace: UpdateEnvironment ruang finspace: UpdateKxClusterCodeConfiguration ruang finspace: UpdateKxClusterDatabases ruang finspace: UpdateKxDatabase ruang finspace: UpdateKxDataview ruang finspace: UpdateKxEnvironment ruang finspace: UpdateKxEnvironmentNetwork ruang finspace: UpdateKxUser ruang finspace: UpdateKxVolume ruang finspace: UpdateUser
firehose	selang api: CreateDeliveryStream selang api: DeleteDeliveryStream selang api: DescribeDeliveryStream selang api: ListDeliveryStreams selang api: StartDeliveryStreamEncryption selang api: StopDeliveryStreamEncryption selang api: UpdateDestination

Awalan layanan	Tindakan
fis	fis: CreateExperimentTemplate fis: CreateTargetAccountConfiguration fis: DeleteExperimentTemplate fis: DeleteTargetAccountConfiguration fis: GetAction fis: GetExperiment fis: GetExperimentTargetAccountConfiguration fis: GetExperimentTemplate fis: GetSafetyLever fis: GetTargetAccountConfiguration fis: GetTargetResourceType fis: ListActions fis: ListExperimentResolvedTargets fis: ListExperimentTargetAccountConfigurations fis: ListExperimentTemplates fis: ListExperiments fis: ListTargetAccountConfigurations fis: ListTargetResourceTypes fis: StartExperiment fis: StopExperiment fis: UpdateExperimentTemplate

Awalan layanan	Tindakan
	fis: UpdateSafetyLeverState fis: UpdateTargetAccountConfiguration

Awalan layanan	Tindakan
fms	fms: AssociateAdminAccount
	fms: AssociateThirdPartyFirewall
	fms: BatchAssociateResource
	fms: BatchDisassociateResource
	fms: DeleteAppsList
	fms: DeleteNotificationChannel
	fms: DeletePolicy
	fms: DeleteProtocolsList
	fms: DeleteResourceSet
	fms: DisassociateAdminAccount
	fms: DisassociateThirdPartyFirewall
	fms: GetAdminAccount
	fms: GetAdminScope
	fms: GetAppsList
	fms: GetComplianceDetail
	fms: GetNotificationChannel
	fms: GetPolicy
	fms: GetProtectionStatus
	fms: GetProtocolsList
	fms: GetResourceSet
	fms: GetThirdPartyFirewallAssociationStatus

Awalan layanan	Tindakan
	fms: GetViolationDetails
	fms: ListAdminAccountsForOrganization
	fms: ListAdminsManagingAccount
	fms: ListAppsLists
	fms: ListComplianceStatus
	fms: ListDiscoveredResources
	fms: ListMemberAccounts
	fms: ListPolicies
	fms: ListProtocolsLists
	fms: ListResourceSetResources
	fms: ListResourceSets
	fms: ListThirdPartyFirewallFirewallPolicies
	fms: PutAdminAccount
	fms: PutAppsList
	fms: PutNotificationChannel
	fms: PutPolicy
	fms: PutProtocolsList
	fms: PutResourceSet

Awalan layanan	Tindakan
detektor penipuan	detektor penipuan: BatchCreateVariable detektor penipuan: BatchGetVariable detektor penipuan: CancelBatchImportJob detektor penipuan: CancelBatchPredictionJob detektor penipuan: CreateBatchImportJob detektor penipuan: CreateBatchPredictionJob detektor penipuan: CreateDetectorVersion detektor penipuan: CreateList detektor penipuan: CreateModel detektor penipuan: CreateModelVersion detektor penipuan: CreateRule detektor penipuan: CreateVariable detektor penipuan: DeleteBatchImportJob detektor penipuan: DeleteBatchPredictionJob detektor penipuan: DeleteDetector detektor penipuan: DeleteDetectorVersion detektor penipuan: DeleteEntityType detektor penipuan: DeleteEvent detektor penipuan: DeleteEventType detektor penipuan: DeleteEventsByEventType detektor penipuan: DeleteExternalModel

Awalan layanan	Tindakan
	detektor penipuan: DeleteLabel
	detektor penipuan: DeleteList
	detektor penipuan: DeleteModel
	detektor penipuan: DeleteModelVersion
	detektor penipuan: DeleteOutcome
	detektor penipuan: DeleteRule
	detektor penipuan: DeleteVariable
	detektor penipuan: DescribeDetector
	detektor penipuan: DescribeModelVersions
	detektor penipuan: GetBatchImportJobs
	detektor penipuan: GetBatchPredictionJobs
	detektor penipuan: GetDeleteEventsByEventTypeStatus
	detektor penipuan: GetDetectorVersion
	detektor penipuan: GetDetectors
	detektor penipuan: GetEntityTypes
	detektor penipuan: GetEvent
	detektor penipuan: GetEventPrediction
	detektor penipuan: GetEventPredictionMetadata
	detektor penipuan: GetEventTypes
	detektor penipuan: GetExternalModels
	detektor penipuan: GetKMSEncryptionKey

Awalan layanan	Tindakan
	detektor penipuan: GetLabels detektor penipuan: GetListElements detektor penipuan: GetListsMetadata detektor penipuan: GetModelVersion detektor penipuan: GetModels detektor penipuan: GetOutcomes detektor penipuan: GetRules detektor penipuan: GetVariables detektor penipuan: ListEventPredictions detektor penipuan: PutDetector detektor penipuan: PutEntityType detektor penipuan: PutEventType detektor penipuan: PutExternalModel detektor penipuan: PutKMSEncryptionKey detektor penipuan: PutLabel detektor penipuan: PutOutcome detektor penipuan: SendEvent detektor penipuan: UpdateDetectorVersion detektor penipuan: UpdateDetectorVersionMetadata detektor penipuan: UpdateDetectorVersionStatus detektor penipuan: UpdateEventLabel

Awalan layanan	Tindakan
	detektor penipuan: UpdateList detektor penipuan: UpdateModel detektor penipuan: UpdateModelVersion detektor penipuan: UpdateModelVersionStatus detektor penipuan: UpdateRuleMetadata detektor penipuan: UpdateRuleVersion detektor penipuan: UpdateVariable

Awalan layanan	Tindakan
fsx	fsx: AssociateFileSystemAliases
	fsx: CancelDataRepositoryTask
	fsx: CopyBackup
	fsx: CreateDataRepositoryTask
	fsx: CreateFileCache
	fsx: CreateFileSystem
	fsx: CreateFileSystemFromBackup
	fsx: CreateSnapshot
	fsx: CreateStorageVirtualMachine
	fsx: CreateVolume
	fsx: CreateVolumeFromBackup
	fsx: DeleteBackup
	fsx: DeleteFileCache
	fsx: DeleteFileSystem
	fsx: DeleteSnapshot
	fsx: DeleteStorageVirtualMachine
	fsx: DeleteVolume
	fsx: DescribeBackups
	fsx: DescribeDataRepositoryAssociations
	fsx: DescribeDataRepositoryTasks
	fsx: DescribeFileCaches

Awalan layanan	Tindakan
	fsx: DescribeFileSystemAliases
	fsx: DescribeFileSystems
	fsx: DescribeS3AccessPointAttachments
	fsx: DescribeSharedVpcConfiguration
	fsx: DescribeSnapshots
	fsx: DescribeStorageVirtualMachines
	fsx: DescribeVolumes
	fsx: DetachAndDeleteS3AccessPoint
	fsx: DisassociateFileSystemAliases
	fsx: ReleaseFileSystemNfsV3Locks
	fsx: RestoreVolumeFromSnapshot
	fsx: StartMisconfiguredStateRecovery
	fsx: UpdateDataRepositoryAssociation
	fsx: UpdateFileCache
	fsx: UpdateFileSystem
	fsx: UpdateSharedVpcConfiguration
	fsx: UpdateSnapshot
	fsx: UpdateStorageVirtualMachine
	fsx: UpdateVolume

Awalan layanan	Tindakan
gamelift	gamelift: AcceptMatch gamelift: ClaimGameServer gamelift: CreateAlias gamelift: CreateBuild gamelift: CreateContainerGroupDefinition gamelift: CreateFleet gamelift: CreateFleetLocations gamelift: CreateGameServerGroup gamelift: CreateGameSession gamelift: CreateGameSessionQueue gamelift: CreateLocation gamelift: CreateMatchmakingConfiguration gamelift: CreateMatchmakingRuleSet gamelift: CreatePlayerSession gamelift: CreatePlayerSessions gamelift: CreateScript gamelift: CreateVpcPeeringAuthorization gamelift: CreateVpcPeeringConnection gamelift: DeleteAlias gamelift: DeleteBuild gamelift: DeleteContainerGroupDefinition

Awalan layanan	Tindakan
	gamelift: DeleteFleet gamelift: DeleteFleetLocations gamelift: DeleteGameServerGroup gamelift: DeleteGameSessionQueue gamelift: DeleteLocation gamelift: DeleteMatchmakingConfiguration gamelift: DeleteMatchmakingRuleSet gamelift: DeleteScalingPolicy gamelift: DeleteScript gamelift: DeleteVpcPeeringAuthorization gamelift: DeleteVpcPeeringConnection gamelift: DeregisterCompute gamelift: DeregisterGameServer gamelift: DescribeAlias gamelift: DescribeBuild gamelift: DescribeCompute gamelift: DescribeContainerFleet gamelift: DescribeContainerGroupDefinition gamelift: DescribeContainerGroupPortMappings gamelift: DescribeEC2InstanceLimits gamelift: DescribeFleetAttributes

Awalan layanan	Tindakan
	gamelift: DescribeFleetCapacity gamelift: DescribeFleetEvents gamelift: DescribeFleetLocationAttributes gamelift: DescribeFleetLocationCapacity gamelift: DescribeFleetLocationUtilization gamelift: DescribeFleetPortSettings gamelift: DescribeFleetUtilization gamelift: DescribeGameServer gamelift: DescribeGameServerGroup gamelift: DescribeGameServerInstances gamelift: DescribeGameSessionDetails gamelift: DescribeGameSessionPlacement gamelift: DescribeGameSessionQueues gamelift: DescribeGameSessions gamelift: DescribeInstances gamelift: DescribeMatchmaking gamelift: DescribeMatchmakingConfigurations gamelift: DescribeMatchmakingRuleSets gamelift: DescribePlayerSessions gamelift: DescribeRuntimeConfiguration gamelift: DescribeScalingPolicies

Awalan layanan	Tindakan
	gamelift: DescribeScript gamelift: DescribeVpcPeeringAuthorizations gamelift: DescribeVpcPeeringConnections gamelift: GetComputeAccess gamelift: GetComputeAuthToken gamelift: GetGameSessionLogUrl gamelift: GetInstanceAccess gamelift: GetPlayerConnectionDetails gamelift: ListAliases gamelift: ListBuilds gamelift: ListCompute gamelift: ListContainerFleets gamelift: ListContainerGroupDefinitionVersions gamelift: ListContainerGroupDefinitions gamelift: ListFleetDeployments gamelift: ListFleets gamelift: ListGameServerGroups gamelift: ListGameServers gamelift: ListLocations gamelift: ListScripts gamelift: PutScalingPolicy

Awalan layanan	Tindakan
	gamelift: RegisterCompute gamelift: RegisterGameServer gamelift: RequestUploadCredentials gamelift: ResolveAlias gamelift: ResumeGameServerGroup gamelift: SearchGameSessions gamelift: StartFleetActions gamelift: StartGameSessionPlacement gamelift: StartMatchBackfill gamelift: StartMatchmaking gamelift: StopFleetActions gamelift: StopGameSessionPlacement gamelift: StopMatchmaking gamelift: SuspendGameServerGroup gamelift: TerminateGameSession gamelift: UpdateAlias gamelift: UpdateBuild gamelift: UpdateContainerGroupDefinition gamelift: UpdateFleetAttributes gamelift: UpdateFleetCapacity gamelift: UpdateFleetPortSettings

Awalan layanan	Tindakan
	gamelift: UpdateGameServer gamelift: UpdateGameServerGroup gamelift: UpdateGameSession gamelift: UpdateGameSessionQueue gamelift: UpdateMatchmakingConfiguration gamelift: UpdateRuntimeConfiguration gamelift: UpdateScript gamelift: ValidateMatchmakingRuleSet

Awalan layanan	Tindakan
geo	geografis: AssociateTrackerConsumer
	geografis: BatchDeleteDevicePositionHistory
	geografis: BatchDeleteGeofence
	geografis: BatchEvaluateGeofences
	geografis: BatchGetDevicePosition
	geografis: BatchPutGeofence
	geografis: BatchUpdateDevicePosition
	geografis: CalculateRoute
	geografis: CalculateRouteMatrix
	geografis: CancelJob
	geografis: CreateGeofenceCollection
	geografis: CreateMap
	geografis: CreatePlaceIndex
	geografis: CreateRouteCalculator
	geografis: CreateTracker
	geografis: DeleteGeofenceCollection
	geografis: DeleteKey
	geografis: DeleteMap
	geografis: DeletePlaceIndex
	geografis: DeleteRouteCalculator
	geografis: DeleteTracker

Awalan layanan	Tindakan
	geografis: DescribeGeofenceCollection
	geografis: DescribeKey
	geografis: DescribeMap
	geografis: DescribePlaceIndex
	geografis: DescribeRouteCalculator
	geografis: DescribeTracker
	geografis: DisassociateTrackerConsumer
	geografis: ForecastGeofenceEvents
	geografis: GetDevicePosition
	geografis: GetDevicePositionHistory
	geografis: GetGeofence
	geografis: GetJob
	geografis: GetMapGlyphs
	geografis: GetMapSprites
	geografis: GetMapStyleDescriptor
	geografis: GetMapTile
	geografis: GetPlace
	geografis: ListDevicePositions
	geografis: ListGeofenceCollections
	geografis: ListGeofences
	geografis: ListJobs

Awalan layanan	Tindakan
	geografis: ListKeys
	geografis: ListMaps
	geografis: ListPlaceIndexes
	geografis: ListRouteCalculators
	geografis: ListTrackerConsumers
	geografis: ListTrackers
	geografis: PutGeofence
	geografis: SearchPlaceIndexForPosition
	geografis: SearchPlaceIndexForSuggestions
	geografis: SearchPlaceIndexForText
	geografis: UpdateGeofenceCollection
	geografis: UpdateKey
	geografis: UpdateMap
	geografis: UpdatePlaceIndex
	geografis: UpdateRouteCalculator
	geografis: UpdateTracker
	geografis: VerifyDevicePosition

Awalan layanan	Tindakan
gletser	gletser: AbortMultipartUpload gletser: AbortVaultLock gletser: CompleteMultipartUpload gletser: CompleteVaultLock gletser: CreateVault gletser: DeleteArchive gletser: DeleteVault gletser: DeleteVaultAccessPolicy gletser: DeleteVaultNotifications gletser: DescribeJob gletser: DescribeVault gletser: GetDataRetrievalPolicy gletser: GetJobOutput gletser: GetVaultAccessPolicy gletser: GetVaultLock gletser: GetVaultNotifications gletser: InitiateJob gletser: InitiateMultipartUpload gletser: InitiateVaultLock gletser: ListJobs gletser: ListMultipartUploads

Awalan layanan	Tindakan
	gletser: ListParts gletser: ListProvisionedCapacity gletser: ListVaults gletser: PurchaseProvisionedCapacity gletser: SetDataRetrievalPolicy gletser: SetVaultAccessPolicy gletser: SetVaultNotifications gletser: UploadArchive gletser: UploadMultipartPart

Awalan layanan	Tindakan
grafana	grafana: AssociateLicense
	grafana: CreateWorkspace
	grafana: CreateWorkspaceApiKey
	grafana: CreateWorkspaceServiceAccount
	grafana: CreateWorkspaceServiceAccountToken
	grafana: DeleteWorkspace
	grafana: DeleteWorkspaceApiKey
	grafana: DeleteWorkspaceServiceAccount
	grafana: DeleteWorkspaceServiceAccountToken
	grafana: DescribeWorkspace
	grafana: DescribeWorkspaceAuthentication
	grafana: DescribeWorkspaceConfiguration
	grafana: DisassociateLicense
	grafana: ListPermissions
	grafana: ListVersions
	grafana: ListWorkspaceServiceAccountTokens
	grafana: ListWorkspaceServiceAccounts
	grafana: ListWorkspaces
	grafana: UpdatePermissions
	grafana: UpdateWorkspace
	grafana: UpdateWorkspaceAuthentication

Awalan layanan	Tindakan
	grafana: UpdateWorkspaceConfiguration

Awalan layanan	Tindakan
greengrass	greengrass: AssociateRoleToGroup greengrass: AssociateServiceRoleToAccount greengrass: BatchAssociateClientDeviceWithCoreDevice greengrass: BatchDisassociateClientDeviceFromCoreDevice greengrass: CancelDeployment greengrass: CreateComponentVersion greengrass: CreateConnectorDefinition greengrass: CreateConnectorDefinitionVersion greengrass: CreateCoreDefinition greengrass: CreateCoreDefinitionVersion greengrass: CreateDeployment greengrass: CreateDeviceDefinition greengrass: CreateDeviceDefinitionVersion greengrass: CreateFunctionDefinition greengrass: CreateFunctionDefinitionVersion greengrass: CreateGroup greengrass: CreateGroupCertificateAuthority greengrass: CreateGroupVersion greengrass: CreateLoggerDefinition greengrass: CreateLoggerDefinitionVersion greengrass: CreateResourceDefinition

Awalan layanan	Tindakan
	greengrass: CreateResourceDefinitionVersion greengrass: CreateSoftwareUpdateJob greengrass: CreateSubscriptionDefinition greengrass: CreateSubscriptionDefinitionVersion greengrass: DeleteComponent greengrass: DeleteConnectorDefinition greengrass: DeleteCoreDefinition greengrass: DeleteCoreDevice greengrass: DeleteDeployment greengrass: DeleteDeviceDefinition greengrass: DeleteFunctionDefinition greengrass: DeleteGroup greengrass: DeleteLoggerDefinition greengrass: DeleteResourceDefinition greengrass: DeleteSubscriptionDefinition greengrass: DescribeComponent greengrass: DisassociateRoleFromGroup greengrass: DisassociateServiceRoleFromAccount greengrass: GetAssociatedRole greengrass: GetBulkDeploymentStatus greengrass: GetComponent

Awalan layanan	Tindakan
	greengrass: GetComponentVersionArtifact greengrass: GetConnectivityInfo greengrass: GetConnectorDefinition greengrass: GetConnectorDefinitionVersion greengrass: GetCoreDefinition greengrass: GetCoreDefinitionVersion greengrass: GetCoreDevice greengrass: GetDeployment greengrass: GetDeploymentStatus greengrass: GetDeviceDefinition greengrass: GetDeviceDefinitionVersion greengrass: GetFunctionDefinition greengrass: GetFunctionDefinitionVersion greengrass: GetGroup greengrass: GetGroupCertificateAuthority greengrass: GetGroupCertificateConfiguration greengrass: GetGroupVersion greengrass: GetLoggerDefinition greengrass: GetLoggerDefinitionVersion greengrass: GetResourceDefinition greengrass: GetResourceDefinitionVersion

Awalan layanan	Tindakan
	greengrass: GetServiceRoleForAccount greengrass: GetSubscriptionDefinition greengrass: GetSubscriptionDefinitionVersion greengrass: GetThingRuntimeConfiguration greengrass: ListBulkDeploymentDetailedReports greengrass: ListBulkDeployments greengrass: ListClientDevicesAssociatedWithCoreDevice greengrass: ListComponentVersions greengrass: ListComponents greengrass: ListConnectorDefinitionVersions greengrass: ListConnectorDefinitions greengrass: ListCoreDefinitionVersions greengrass: ListCoreDefinitions greengrass: ListCoreDevices greengrass: ListDeployments greengrass: ListDeviceDefinitionVersions greengrass: ListDeviceDefinitions greengrass: ListEffectiveDeployments greengrass: ListFunctionDefinitionVersions greengrass: ListFunctionDefinitions greengrass: ListGroupCertificateAuthorities

Awalan layanan	Tindakan
	greengrass: ListGroupVersions greengrass: ListGroups greengrass: ListInstalledComponents greengrass: ListLoggerDefinitionVersions greengrass: ListLoggerDefinitions greengrass: ListResourceDefinitionVersions greengrass: ListResourceDefinitions greengrass: ListSubscriptionDefinitionVersions greengrass: ListSubscriptionDefinitions greengrass: ResetDeployments greengrass: StartBulkDeployment greengrass: StopBulkDeployment greengrass: UpdateConnectivityInfo greengrass: UpdateConnectorDefinition greengrass: UpdateCoreDefinition greengrass: UpdateDeviceDefinition greengrass: UpdateFunctionDefinition greengrass: UpdateGroup greengrass: UpdateGroupCertificateConfiguration greengrass: UpdateLoggerDefinition greengrass: UpdateResourceDefinition

Awalan layanan	Tindakan
	greengrass: UpdateSubscriptionDefinition greengrass: UpdateThingRuntimeConfiguration

Awalan layanan	Tindakan
stasiun tanah	stasiun tanah: CancelContact
	stasiun tanah: CreateConfig
	stasiun tanah: CreateDataflowEndpointGroup
	stasiun tanah: CreateDataflowEndpointGroupV2
	stasiun tanah: CreateEphemeris
	stasiun tanah: CreateMissionProfile
	stasiun tanah: DeleteConfig
	stasiun tanah: DeleteDataflowEndpointGroup
	stasiun tanah: DeleteEphemeris
	stasiun tanah: DeleteMissionProfile
	stasiun tanah: DescribeContact
	stasiun tanah: DescribeContactVersion
	stasiun tanah: DescribeEphemeris
	stasiun tanah: GetConfig
	stasiun tanah: GetDataflowEndpointGroup
	stasiun tanah: GetMinuteUsage
	stasiun tanah: GetMissionProfile
	stasiun tanah: GetSatellite
	stasiun tanah: ListAntennas
	stasiun tanah: ListConfigs
	stasiun tanah: ListContacts

Awalan layanan	Tindakan
	stasiun tanah: ListDataflowEndpointGroups
	stasiun tanah: ListEphemerides
	stasiun tanah: ListGroundStationReservations
	stasiun tanah: ListGroundStations
	stasiun tanah: ListMissionProfiles
	stasiun tanah: ListSatellites
	stasiun tanah: RegisterAgent
	stasiun tanah: ReserveContact
	stasiun tanah: UpdateAgentStatus
	stasiun tanah: UpdateConfig
	stasiun tanah: UpdateEphemeris
	stasiun tanah: UpdateMissionProfile

Awalan layanan	Tindakan
tugas jaga	tugas jaga: AcceptAdministratorInvitation tugas jaga: AcceptInvitation tugas jaga: ArchiveFindings tugas jaga: CreateDetector tugas jaga: CreateFilter GuardDuty:CreateIPset tugas jaga: CreateInvestigation tugas jaga: CreateMalwareProtectionPlan tugas jaga: CreateMembers tugas jaga: CreatePublishingDestination tugas jaga: CreateSampleFindings tugas jaga: CreateThreatEntitySet tugas jaga: CreateThreatIntelSet tugas jaga: CreateTrustedEntitySet tugas jaga: DeclineInvitations tugas jaga: DeleteDetector tugas jaga: DeleteFilter guardDuty:deleteIPset tugas jaga: DeleteInvitations tugas jaga: DeleteMalwareProtectionPlan tugas jaga: DeleteMembers

Awalan layanan	Tindakan
	tugas jaga: DeletePublishingDestination
	tugas jaga: DeleteThreatEntitySet
	tugas jaga: DeleteThreatIntelSet
	tugas jaga: DeleteTrustedEntitySet
	tugas jaga: DescribeMalwareScans
	tugas jaga: DescribeOrganizationConfiguration
	tugas jaga: DescribePublishingDestination
	tugas jaga: DisableOrganizationAdminAccount
	tugas jaga: DisassociateFromAdministratorAccount
	tugas jaga: DisassociateFromMasterAccount
	tugas jaga: DisassociateMembers
	tugas jaga: EnableOrganizationAdminAccount
	tugas jaga: GetAdministratorAccount
	tugas jaga: GetCoverageStatistics
	tugas jaga: GetDetector
	tugas jaga: GetFilter
	tugas jaga: GetFindings
	tugas jaga: GetFindingsStatistics
	GuardDuty:GETIPSET
	tugas jaga: GetInvestigation
	tugas jaga: GetInvitationsCount

Awalan layanan	Tindakan
	tugas jaga: GetMalwareProtectionPlan tugas jaga: GetMalwareScan tugas jaga: GetMalwareScanSettings tugas jaga: GetMasterAccount tugas jaga: GetMemberDetectors tugas jaga: GetMembers tugas jaga: GetOrganizationStatistics tugas jaga: GetRemainingFreeTrialDays tugas jaga: GetThreatEntitySet tugas jaga: GetThreatIntelSet tugas jaga: GetTrustedEntitySet tugas jaga: GetUsageStatistics tugas jaga: InviteMembers tugas jaga: ListCoverage tugas jaga: ListDetectors tugas jaga: ListFilters tugas jaga: ListFindings guardduty:Listipsets tugas jaga: ListInvestigations tugas jaga: ListInvitations tugas jaga: ListMalwareProtectionPlans

Awalan layanan	Tindakan
	tugas jaga: ListMalwareScans tugas jaga: ListMembers tugas jaga: ListOrganizationAdminAccounts tugas jaga: ListPublishingDestinations tugas jaga: ListThreatEntitySets tugas jaga: ListThreatIntelSets tugas jaga: ListTrustedEntitySets tugas jaga: StartMalwareScan tugas jaga: StartMonitoringMembers tugas jaga: StopMonitoringMembers tugas jaga: UnarchiveFindings tugas jaga: UpdateDetector tugas jaga: UpdateFilter tugas jaga: UpdateFindingsFeedback GuardDuty: UpdateIPSET tugas jaga: UpdateMalwareProtectionPlan tugas jaga: UpdateMalwareScanSettings tugas jaga: UpdateMemberDetectors tugas jaga: UpdateOrganizationConfiguration tugas jaga: UpdatePublishingDestination tugas jaga: UpdateThreatEntitySet

Awalan layanan	Tindakan
	tugas jaga: UpdateThreatIntelSet tugas jaga: UpdateTrustedEntitySet

Awalan layanan	Tindakan
healthlake	Healthlake: CancelFHIRExportJobWithDelete
	Healthlake: CreateDataTransformationProfile
	Healthlake:CreateFhirDataStore
	Healthlake: CreateResource
	Healthlake: DeleteDataTransformationProfile
	HealthLake:DeleteFhirDataStore
	Healthlake: DeleteResource
	Healthlake: DescribeDataTransformationJob
	Healthlake:DescribefhirDataStore
	Healthlake: DescribeFHIRExportJob
	Healthlake: DescribeFHIRExportJobWithGet
	Healthlake: DescribeFHIRImportJob
	Healthlake: GetCapabilities
	Healthlake: GetDataTransformationProfile
	Healthlake: ListDataTransformationJobs
	Healthlake: ListDataTransformationProfileVersions
	Healthlake: ListDataTransformationProfiles
	HealthLake:ListfHirDataStore
	Healthlake: ListFHIRExportJobs
	Healthlake: ListFHIRImportJobs
	Healthlake: PublishDataTransformationProfile

Awalan layanan	Tindakan
	Healthlake: ReadResource
	Healthlake: SearchEverything
	Healthlake: SearchWithGet
	Healthlake: SearchWithPost
	Healthlake: StartFHIRExportJob
	Healthlake: StartFHIRExportJobWithPost
	Healthlake: StartFHIRImportJob
	Healthlake: UpdateDataTransformationProfile
	HealthLake:UpdateFhirDataStore
	Healthlake: UpdateProfileWithAgent
	Healthlake: UpdateResource

Awalan layanan	Tindakan
kode madu	kode madu: BatchCreateTableRows kode madu: BatchDeleteTableRows kode madu: BatchUpdateTableRows kode madu: BatchUpsertTableRows kode madu: DescribeTableDataImportJob kode madu: GetScreenData kode madu: InvokeScreenAutomation kode madu: ListTableColumns kode madu: ListTableRows kode madu: ListTables kode madu: QueryTableRows kode madu: StartTableDataImportJob

Awalan layanan	Tindakan
iam	saya: AddClientIDToOpenIDConnectProvider
	saya: AddRoleToInstanceProfile
	saya: AddUserToGroup
	saya: AttachGroupPolicy
	saya: AttachRolePolicy
	saya: AttachUserPolicy
	saya: ChangePassword
	saya: CreateAccessKey
	saya: CreateAccountAlias
	saya: CreateGroup
	saya: CreateInstanceProfile
	saya: CreateLoginProfile
	saya: CreateOpenIDConnectProvider
	saya: CreatePolicy
	saya: CreatePolicyVersion
	saya: CreateRole
	iam:CreateSamlProvider
	saya: CreateServiceLinkedRole
	saya: CreateServiceSpecificCredential
	saya: CreateUser
	saya: CreateVirtualMFADevice

Awalan layanan	Tindakan
	iam:Nonaktifkan FADEVICE
	saya: DeleteAccessKey
	saya: DeleteAccountAlias
	saya: DeleteAccountPasswordPolicy
	saya: DeleteCloudFrontPublicKey
	saya: DeleteGroup
	saya: DeleteGroupPolicy
	saya: DeleteInstanceProfile
	saya: DeleteLoginProfile
	saya: DeleteOpenIDConnectProvider
	saya: DeletePolicy
	saya: DeletePolicyVersion
	saya: DeleteRole
	saya: DeleteRolePermissionsBoundary
	saya: DeleteRolePolicy
	Iam:deleteCamlProvider
	saya: DeleteSSHPublicKey
	saya: DeleteServerCertificate
	saya: DeleteServiceLinkedRole
	saya: DeleteServiceSpecificCredential
	saya: DeleteSigningCertificate

Awalan layanan	Tindakan
	saya: DeleteUser
	saya: DeleteUserPermissionsBoundary
	saya: DeleteUserPolicy
	saya: DeleteVirtualMFADevice
	saya: DetachGroupPolicy
	saya: DetachRolePolicy
	saya: DetachUserPolicy
	saya: DisableOrganizationsRootCredentialsManagement
	saya: DisableOrganizationsRootSessions
	saya: DisableOutboundWebIdentityFederation
	Iam:enableMfadevice
	saya: EnableOrganizationsRootCredentialsManagement
	saya: EnableOrganizationsRootSessions
	saya: EnableOutboundWebIdentityFederation
	saya: GenerateCredentialReport
	saya: GenerateOrganizationsAccessReport
	saya: GenerateServiceLastAccessedDetails
	saya: GetAccessKeyLastUsed
	saya: GetAccountAuthorizationDetails
	saya: GetAccountEmailAddress
	saya: GetAccountName

Awalan layanan	Tindakan
	saya: GetAccountPasswordPolicy
	saya: GetAccountSummary
	saya: GetCloudFrontPublicKey
	saya: GetContextKeysForCustomPolicy
	saya: GetContextKeysForPrincipalPolicy
	saya: GetCredentialReport
	saya: GetGroup
	saya: GetGroupPolicy
	saya: GetInstanceProfile
	saya: GetLoginProfile
	Iam:GetMFADevice
	saya: GetOpenIDConnectProvider
	saya: GetOrganizationsAccessReport
	saya: GetOutboundWebIdentityFederationInfo
	saya: GetPolicy
	saya: GetPolicyVersion
	saya: GetRole
	saya: GetRolePolicy
	iam:GetSamlProvider
	saya: GetSSHPublicKey
	saya: GetServerCertificate

Awalan layanan	Tindakan
	saya: GetServiceLastAccessedDetails
	saya: GetServiceLastAccessedDetailsWithEntities
	saya: GetServiceLinkedRoleDeletionStatus
	saya: GetUser
	saya: GetUserPolicy
	saya: ListAccessKeys
	saya: ListAccountAliases
	saya: ListAttachedGroupPolicies
	saya: ListAttachedRolePolicies
	saya: ListAttachedUserPolicies
	saya: ListCloudFrontPublicKeys
	saya: ListDelegationRequests
	saya: ListEntitiesForPolicy
	saya: ListGroupPolicies
	saya: ListGroups
	saya: ListGroupsForUser
	saya: ListInstanceProfiles
	saya: ListInstanceProfilesForRole
	Iam:ListmFADevices
	saya: ListOpenIDConnectProviders
	saya: ListOrganizationsFeatures

Awalan layanan	Tindakan
	saya: ListPolicies
	saya: ListPoliciesGrantingServiceAccess
	saya: ListPolicyVersions
	saya: ListRolePolicies
	saya: ListRoles
	iam:ListSamlProviders
	saya: ListSSHPublicKeys
	saya: ListSTSRegionalEndpointsStatus
	saya: ListServerCertificates
	saya: ListServiceSpecificCredentials
	saya: ListSigningCertificates
	saya: ListUserPolicies
	saya: ListUsers
	saya: ListVirtualMFADevices
	saya: PutGroupPolicy
	saya: PutRolePermissionsBoundary
	saya: PutRolePolicy
	saya: PutUserPermissionsBoundary
	saya: PutUserPolicy
	saya: RemoveClientIDFromOpenIDConnectProvider
	saya: RemoveRoleFromInstanceProfile

Awalan layanan	Tindakan
	saya: RemoveUserFromGroup
	saya: ResetServiceSpecificCredential
	iam:resyncmfadevice
	saya: SetDefaultPolicyVersion
	saya: SetSTSRegionalEndpointStatus
	saya: SetSecurityTokenServicePreferences
	saya: SimulateCustomPolicy
	saya: SimulatePrincipalPolicy
	saya: UpdateAccessKey
	saya: UpdateAccountEmailAddress
	saya: UpdateAccountName
	saya: UpdateAccountPasswordPolicy
	saya: UpdateAssumeRolePolicy
	saya: UpdateCloudFrontPublicKey
	saya: UpdateGroup
	saya: UpdateLoginProfile
	saya: UpdateOpenIDConnectProviderThumbprint
	saya: UpdateRole
	saya: UpdateRoleDescription
	iam:UpdateSamlProvider
	saya: UpdateSSHPublicKey

Awalan layanan	Tindakan
	saya: UpdateServerCertificate saya: UpdateServiceSpecificCredential saya: UpdateSigningCertificate saya: UpdateUser saya: UploadCloudFrontPublicKey saya: UploadSSHPublicKey saya: UploadServerCertificate saya: UploadSigningCertificate

Awalan layanan	Tindakan
toko identitas	toko identitas: CreateGroup toko identitas: CreateGroupMembership toko identitas: CreateUser toko identitas: DeleteGroup toko identitas: DeleteGroupMembership toko identitas: DeleteUser toko identitas: DescribeGroup toko identitas: DescribeGroupMembership toko identitas: DescribeUser toko identitas: GetGroupId toko identitas: GetGroupMembershipId toko identitas: GetUserId toko identitas: IsMemberInGroups toko identitas: ListGroupMemberships toko identitas: ListGroupMembershipsForMember toko identitas: ListGroups toko identitas: ListUsers toko identitas: UpdateGroup toko identitas: UpdateUser

Awalan layanan	Tindakan
imagebuilder	imagebuilder: CancelImageCreation
	imagebuilder: CancelLifecycleExecution
	imagebuilder: CreateComponent
	imagebuilder: CreateContainerRecipe
	imagebuilder: CreateDistributionConfiguration
	imagebuilder: CreateImage
	imagebuilder: CreateImagePipeline
	imagebuilder: CreateImageRecipe
	imagebuilder: CreateInfrastructureConfiguration
	imagebuilder: CreateLifecyclePolicy
	imagebuilder: CreateWorkflow
	imagebuilder: DeleteComponent
	imagebuilder: DeleteContainerRecipe
	imagebuilder: DeleteDistributionConfiguration
	imagebuilder: DeleteImage
	imagebuilder: DeleteImagePipeline
	imagebuilder: DeleteImageRecipe
	imagebuilder: DeleteInfrastructureConfiguration
	imagebuilder: DeleteLifecyclePolicy
	imagebuilder: DeleteWorkflow
	imagebuilder: DistributeImage

Awalan layanan	Tindakan
	imagebuilder: GetComponentPolicy
	imagebuilder: GetContainerRecipePolicy
	imagebuilder: GetImagePolicy
	imagebuilder: GetImageRecipePolicy
	imagebuilder: GetLifecycleExecution
	imagebuilder: GetLifecyclePolicy
	imagebuilder: GetMarketplaceResource
	imagebuilder: GetWorkflowExecution
	imagebuilder: GetWorkflowStepExecution
	imagebuilder: ImportComponent
	imagebuilder: ImportDiskImage
	imagebuilder: ImportVmImage
	imagebuilder: ListComponentBuildVersions
	imagebuilder: ListComponents
	imagebuilder: ListContainerRecipes
	imagebuilder: ListDistributionConfigurations
	imagebuilder: ListImageBuildVersions
	imagebuilder: ListImagePackages
	imagebuilder: ListImagePipelineImages
	imagebuilder: ListImagePipelines
	imagebuilder: ListImageRecipes

Awalan layanan	Tindakan
	imagebuilder: ListImageScanFindingAggregations
	imagebuilder: ListImageScanFindings
	imagebuilder: ListImages
	imagebuilder: ListInfrastructureConfigurations
	imagebuilder: ListLifecycleExecutionResources
	imagebuilder: ListLifecycleExecutions
	imagebuilder: ListLifecyclePolicies
	imagebuilder: ListWaitingWorkflowSteps
	imagebuilder: ListWorkflowExecutions
	imagebuilder: ListWorkflowStepExecutions
	imagebuilder: ListWorkflows
	imagebuilder: PutComponentPolicy
	imagebuilder: PutContainerRecipePolicy
	imagebuilder: PutImagePolicy
	imagebuilder: PutImageRecipePolicy
	imagebuilder: RetryImage
	imagebuilder: SendWorkflowStepAction
	imagebuilder: StartImagePipelineExecution
	imagebuilder: StartResourceStateUpdate
	imagebuilder: UpdateDistributionConfiguration
	imagebuilder: UpdateImagePipeline

Awalan layanan	Tindakan
	imagebuilder: UpdateInfrastructureConfiguration

Awalan layanan	Tindakan
inspektur	inspektur: AddAttributesToFindings inspektur: CreateAssessmentTarget inspektur: CreateAssessmentTemplate inspektur: CreateExclusionsPreview inspektur: CreateResourceGroup inspektur: DeleteAssessmentRun inspektur: DeleteAssessmentTarget inspektur: DeleteAssessmentTemplate inspektur: DescribeAssessmentRuns inspektur: DescribeAssessmentTargets inspektur: DescribeAssessmentTemplates inspektur: DescribeCrossAccountAccessRole inspektur: DescribeExclusions inspektur: DescribeFindings inspektur: DescribeResourceGroups inspektur: DescribeRulesPackages inspektur: GetAssessmentReport inspektur: GetExclusionsPreview inspektur: GetTelemetryMetadata inspektur: ListAssessmentRunAgents inspektur: ListAssessmentRuns

Awalan layanan	Tindakan
	inspektur: ListAssessmentTargets inspektur: ListAssessmentTemplates inspektur: ListEventSubscriptions inspektur: ListExclusions inspektur: ListFindings inspektur: ListRulesPackages inspektur: PreviewAgents inspektur: RegisterCrossAccountAccessRole inspektur: RemoveAttributesFromFindings inspektur: StartAssessmentRun inspektur: StopAssessmentRun inspektur: SubscribeToEvent inspektur: UnsubscribeFromEvent inspektur: UpdateAssessmentTarget

Awalan layanan	Tindakan
inspektor2	inspektor2: AssociateMember
	inspektor2: BatchGetAccountStatus
	inspektor2: BatchGetCodeSnippet
	inspektor2: BatchGetFindingDetails
	inspektor2: BatchGetFreeTrialInfo
	inspektor2: BatchGetMemberEc2DeepInspectionStatus
	inspektor2: BatchUpdateMemberEc2DeepInspectionStatus
	inspektor2: CancelFindingsReport
	inspektor2: CancelSbomExport
	inspektor2: CreateCisScanConfiguration
	inspektor2: CreateCodeSecurityIntegration
	inspektor2: CreateConnector
	inspektor2: CreateFilter
	inspektor2: CreateFindingsReport
	inspektor2: CreateSbomExport
	inspektor2: DeleteCisScanConfiguration
	inspektor2: DeleteCodeSecurityIntegration
	inspektor2: DeleteConnector
	inspektor2: DeleteFilter
	inspektor2: DescribeOrganizationConfiguration
	Inspector2: Nonaktifkan

Awalan layanan	Tindakan
	inspektor2: DisableDelegatedAdminAccount
	inspektor2: DisassociateMember
	Inspector2: aktifkan
	inspektor2: EnableDelegatedAdminAccount
	inspektor2: GetCisScanReport
	inspektor2: GetCisScanResultDetails
	inspektor2: GetClustersForImage
	inspektor2: GetCodeSecurityIntegration
	inspektor2: GetCodeSecurityScan
	inspektor2: GetConfiguration
	inspektor2: GetDelegatedAdminAccount
	inspektor2: GetEc2DeepInspectionConfiguration
	inspektor2: GetEncryptionKey
	inspektor2: GetFindingsReportStatus
	inspektor2: GetMember
	inspektor2: GetSbomExport
	inspektor2: ListAccountPermissions
	inspektor2: ListCisScanConfigurations
	inspektor2: ListCisScanResultsAggregatedByChecks
	inspektor2: ListCisScanResultsAggregatedByTargetResource
	inspektor2: ListCisScans

Awalan layanan	Tindakan
	inspektor2: ListCodeSecurityIntegrations
	inspektor2: ListCodeSecurityScanConfigurations
	inspektor2: ListConnectorScanConfigurations
	inspektor2: ListConnectors
	inspektor2: ListCoverage
	inspektor2: ListCoverageStatistics
	inspektor2: ListDelegatedAdminAccounts
	inspektor2: ListFilters
	inspektor2: ListFindingAggregations
	inspektor2: ListFindings
	inspektor2: ListMembers
	inspektor2: ListUsageTotals
	inspektor2: ResetEncryptionKey
	inspektor2: SearchVulnerabilities
	inspektor2: SendCisSessionHealth
	inspektor2: SendCisSessionTelemetry
	inspektor2: StartCisSession
	inspektor2: StartCodeSecurityScan
	inspektor2: StopCisSession
	inspektor2: UpdateCisScanConfiguration
	inspektor2: UpdateCodeSecurityIntegration

Awalan layanan	Tindakan
	inspektor2: UpdateConfiguration
	inspektor2: UpdateConnector
	inspektor2: UpdateEc2DeepInspectionConfiguration
	inspektor2: UpdateEncryptionKey
	inspektor2: UpdateFilter
	inspektor2: UpdateOrgEc2DeepInspectionConfiguration
	inspektor2: UpdateOrganizationConfiguration

Awalan layanan	Tindakan
iot	IOT: AcceptCertificateTransfer
	IOT: AddThingToBillingGroup
	IOT: AddThingToThingGroup
	IOT: AssociateSbomWithPackageVersion
	IOT: AssociateTargetsWithJob
	IOT: AttachPolicy
	IOT: AttachPrincipalPolicy
	IOT: AttachSecurityProfile
	IOT: AttachThingPrincipal
	IOT: CancelAuditMitigationActionsTask
	IOT: CancelAuditTask
	IOT: CancelCertificateTransfer
	IOT: CancelDetectMitigationActionsTask
	IOT: CancelJob
	IOT: CancelJobExecution
	IOT: ClearDefaultAuthorizer
	IOT: ConfirmTopicRuleDestination
	IOT: CreateAuditSuppression
	IOT: CreateAuthorizer
	IOT: CreateBillingGroup
	IOT: CreateCertificateFromCsr

Awalan layanan	Tindakan
	IOT: CreateCertificateProvider IOT: CreateCommand IOT: CreateCustomMetric IOT: CreateDimension IOT: CreateDomainConfiguration IOT: CreateDynamicThingGroup IOT: CreateFleetMetric IOT: CreateJob IOT: CreateJobTemplate IOT: CreateKeysAndCertificate IOT: CreateMitigationAction IoT: CreateoTaupdate IOT: CreatePackage IOT: CreatePackageVersion IOT: CreatePolicy IOT: CreatePolicyVersion IOT: CreateProvisioningClaim IOT: CreateProvisioningTemplate IOT: CreateProvisioningTemplateVersion IOT: CreateRoleAlias IOT: CreateScheduledAudit

Awalan layanan	Tindakan
	IOT: CreateSecurityProfile IOT: CreateStream IOT: CreateThing IOT: CreateThingGroup IOT: CreateThingType IOT: CreateTopicRule IOT: CreateTopicRuleDestination IOT: DeleteAccountAuditConfiguration IOT: DeleteAuditSuppression IOT: DeleteAuthorizer IOT: DeleteBillingGroup IoT: DeleteCacertificate IOT: DeleteCertificate IOT: DeleteCertificateProvider IOT: DeleteCommand IOT: DeleteCustomMetric IOT: DeleteDimension IOT: DeleteDomainConfiguration IOT: DeleteDynamicThingGroup IOT: DeleteFleetMetric IOT: DeleteJob

Awalan layanan	Tindakan
	IOT: DeleteJobExecution IOT: DeleteJobTemplate IOT: DeleteMitigationAction IoT: DeleteOtaUpdate IOT: DeletePackage IOT: DeletePackageVersion IOT: DeletePolicy IOT: DeletePolicyVersion IOT: DeleteProvisioningTemplate IOT: DeleteProvisioningTemplateVersion IOT: DeleteRegistrationCode IOT: DeleteRoleAlias IOT: DeleteScheduledAudit IOT: DeleteSecurityProfile IOT: DeleteStream IOT: DeleteThing IOT: DeleteThingGroup IOT: DeleteThingType IOT: DeleteTopicRule IOT: DeleteTopicRuleDestination IOT: DeleteV2LoggingLevel

Awalan layanan	Tindakan
	IOT: DeprecateThingType IOT: DescribeAccountAuditConfiguration IOT: DescribeAuditFinding IOT: DescribeAuditMitigationActionsTask IOT: DescribeAuditSuppression IOT: DescribeAuditTask IOT: DescribeAuthorizer IOT: DescribeBillingGroup IoT:Describecacertificate IOT: DescribeCertificate IOT: DescribeCertificateProvider IOT: DescribeCustomMetric IOT: DescribeDefaultAuthorizer IOT: DescribeDetectMitigationActionsTask IOT: DescribeDimension IOT: DescribeDomainConfiguration IOT: DescribeEncryptionConfiguration IOT: DescribeEndpoint IOT: DescribeEventConfigurations IOT: DescribeFleetMetric IOT: DescribeIndex

Awalan layanan	Tindakan
	IOT: DescribeJob IOT: DescribeJobExecution IOT: DescribeJobTemplate IOT: DescribeManagedJobTemplate IOT: DescribeMitigationAction IOT: DescribeProvisioningTemplate IOT: DescribeProvisioningTemplateVersion IOT: DescribeRoleAlias IOT: DescribeScheduledAudit IOT: DescribeSecurityProfile IOT: DescribeStream IOT: DescribeThing IOT: DescribeThingGroup IOT: DescribeThingRegistrationTask IOT: DescribeThingType IOT: DetachPolicy IOT: DetachPrincipalPolicy IOT: DetachSecurityProfile IOT: DetachThingPrincipal IOT: DisableTopicRule IOT: DisassociateSbomFromPackageVersion

Awalan layanan	Tindakan
	IOT: EnableTopicRule
	IOT: GetBehaviorModelTrainingSummaries
	IOT: GetBucketsAggregation
	IOT: GetCardinality
	IOT: GetCommand
	IOT: GetEffectivePolicies
	IOT: GetJobDocument
	IOT: GetLoggingOptions
	IoT: getoTaupdate
	IOT: GetPackage
	IOT: GetPackageConfiguration
	IOT: GetPackageVersion
	IOT: GetPercentiles
	IOT: GetPolicy
	IOT: GetPolicyVersion
	IOT: GetRegistrationCode
	IOT: GetStatistics
	IOT: GetThingConnectivityData
	IOT: GetTopicRule
	IOT: GetTopicRuleDestination
	IOT: GetV2LoggingOptions

Awalan layanan	Tindakan
	IOT: ListActiveViolations IOT: ListAttachedPolicies IOT: ListAuditFindings IOT: ListAuditMitigationActionsExecutions IOT: ListAuditMitigationActionsTasks IOT: ListAuditSuppressions IOT: ListAuditTasks IOT: ListAuthorizers IOT: ListBillingGroups IoT: ListCacertificates IOT: ListCertificateProviders IOT: ListCertificates IOT: ListCertificatesByCA IOT: ListCommands IOT: ListCustomMetrics IOT: ListDetectMitigationActionsExecutions IOT: ListDetectMitigationActionsTasks IOT: ListDimensions IOT: ListDomainConfigurations IOT: ListFleetMetrics IOT: ListIndices

Awalan layanan	Tindakan
	IOT: ListJobExecutionsForJob IOT: ListJobExecutionsForThing IOT: ListJobTemplates IOT: ListJobs IOT: ListManagedJobTemplates IOT: ListMetricValues IOT: ListMitigationActions IoT: ListoTaupdates IOT: ListOutgoingCertificates IOT: ListPackageVersions IOT: ListPackages IOT: ListPolicies IOT: ListPolicyPrincipals IOT: ListPolicyVersions IOT: ListPrincipalPolicies IOT: ListPrincipalThings IOT: ListPrincipalThingsV2 IOT: ListProvisioningTemplateVersions IOT: ListProvisioningTemplates IOT: ListRelatedResourcesForAuditFinding IOT: ListRoleAliases

Awalan layanan	Tindakan
	IOT: ListSbomValidationResults
	IOT: ListScheduledAudits
	IOT: ListSecurityProfiles
	IOT: ListSecurityProfilesForTarget
	IOT: ListStreams
	IOT: ListTargetsForPolicy
	IOT: ListTargetsForSecurityProfile
	IOT: ListThingGroups
	IOT: ListThingGroupsForThing
	IOT: ListThingPrincipals
	IOT: ListThingPrincipalsV2
	IOT: ListThingRegistrationTaskReports
	IOT: ListThingRegistrationTasks
	IOT: ListThingTypes
	IOT: ListThings
	IOT: ListThingsInBillingGroup
	IOT: ListThingsInThingGroup
	IOT: ListTopicRuleDestinations
	IOT: ListTopicRules
	IOT: ListV2LoggingLevels
	IOT: ListViolationEvents

Awalan layanan	Tindakan
	IOT: PutVerificationStateOnViolation IoT: registercacertificate IOT: RegisterCertificate IOT: RegisterCertificateWithoutCA IOT: RegisterThing IOT: RejectCertificateTransfer IOT: RemoveThingFromBillingGroup IOT: RemoveThingFromThingGroup IOT: ReplaceTopicRule IOT: SearchIndex IOT: SetDefaultAuthorizer IOT: SetDefaultPolicyVersion IOT: SetLoggingOptions IOT: SetV2LoggingLevel IOT: SetV2LoggingOptions IOT: StartAuditMitigationActionsTask IOT: StartDetectMitigationActionsTask IOT: StartOnDemandAuditTask IOT: StartThingRegistrationTask IOT: StopThingRegistrationTask IOT: TestAuthorization

Awalan layanan	Tindakan
	IOT: TestInvokeAuthorizer IOT: TransferCertificate IOT: UpdateAccountAuditConfiguration IOT: UpdateAuditSuppression IOT: UpdateAuthorizer IOT: UpdateBillingGroup IoT: UpdateCacertificate IOT: UpdateCertificate IOT: UpdateCertificateProvider IOT: UpdateCommand IOT: UpdateCustomMetric IOT: UpdateDimension IOT: UpdateDomainConfiguration IOT: UpdateDynamicThingGroup IOT: UpdateEncryptionConfiguration IOT: UpdateEventConfigurations IOT: UpdateFleetMetric IOT: UpdateIndexingConfiguration IOT: UpdateJob IOT: UpdateMitigationAction IOT: UpdatePackage

Awalan layanan	Tindakan
	IOT: UpdatePackageConfiguration IOT: UpdatePackageVersion IOT: UpdateProvisioningTemplate IOT: UpdateRoleAlias IOT: UpdateScheduledAudit IOT: UpdateSecurityProfile IOT: UpdateStream IOT: UpdateThing IOT: UpdateThingGroup IOT: UpdateThingGroupsForThing IOT: UpdateThingType IOT: UpdateTopicRuleDestination IOT: ValidateSecurityProfileBehaviors

Awalan layanan	Tindakan
iotanalitik	iotanalitik: CancelPipelineReprocessing
	iotanalitik: CreateChannel
	iotanalitik: CreateDataset
	iotanalitik: CreateDatasetContent
	iotanalitik: CreateDatastore
	iotanalitik: CreatePipeline
	iotanalitik: DeleteChannel
	iotanalitik: DeleteDataset
	iotanalitik: DeleteDatasetContent
	iotanalitik: DeleteDatastore
	iotanalitik: DeletePipeline
	iotanalitik: DescribeChannel
	iotanalitik: DescribeDataset
	iotanalitik: DescribeDatastore
	iotanalitik: DescribeLoggingOptions
	iotanalitik: DescribePipeline
	iotanalitik: GetDatasetContent
	iotanalitik: ListChannels
	iotanalitik: ListDatasetContents
	iotanalitik: ListDatasets
	iotanalitik: ListDatastores

Awalan layanan	Tindakan
	iotanalitik: ListPipelines iotanalitik: PutLoggingOptions iotanalitik: RunPipelineActivity iotanalitik: SampleChannelData iotanalitik: StartPipelineReprocessing iotanalitik: UpdateChannel iotanalitik: UpdateDataset iotanalitik: UpdateDatastore iotanalitik: UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor: CreateSuiteDefinition iotdeviceadvisor: DeleteSuiteDefinition iotdeviceadvisor: GetEndpoint iotdeviceadvisor: GetSuiteDefinition iotdeviceadvisor: GetSuiteRun iotdeviceadvisor: GetSuiteRunReport iotdeviceadvisor: ListSuiteDefinitions iotdeviceadvisor: ListSuiteRuns iotdeviceadvisor: StartSuiteRun iotdeviceadvisor: StopSuiteRun iotdeviceadvisor: UpdateSuiteDefinition

Awalan layanan	Tindakan
iotevents	iotevents: BatchAcknowledgeAlarm
	iotevents: BatchDeleteDetector
	iotevents: BatchDisableAlarm
	iotevents: BatchEnableAlarm
	iotevents: BatchResetAlarm
	iotevents: BatchSnoozeAlarm
	iotevents: BatchUpdateDetector
	iotevents: CreateAlarmModel
	iotevents: CreateDetectorModel
	iotevents: CreateInput
	iotevents: DeleteAlarmModel
	iotevents: DeleteDetectorModel
	iotevents: DeleteInput
	iotevents: DescribeAlarm
	iotevents: DescribeAlarmModel
	iotevents: DescribeDetector
	iotevents: DescribeDetectorModel
	iotevents: DescribeDetectorModelAnalysis
	iotevents: DescribeInput
	iotevents: DescribeLoggingOptions
	iotevents: GetDetectorModelAnalysisResults

Awalan layanan	Tindakan
	iotevents: ListAlarmModelVersions iotevents: ListAlarmModels iotevents: ListAlarms iotevents: ListDetectorModelVersions iotevents: ListDetectorModels iotevents: ListDetectors iotevents: ListInputRoutings iotevents: ListInputs iotevents: PutLoggingOptions iotevents: StartDetectorModelAnalysis iotevents: UpdateAlarmModel iotevents: UpdateDetectorModel iotevents: UpdateInput
iotfleethub	iotfleethub: CreateApplication iotfleethub: DeleteApplication iotfleethub: DescribeApplication iotfleethub: ListApplications iotfleethub: UpdateApplication

Awalan layanan	Tindakan
iotsitewise	iotsitewise: AssociateAssets
	iotsitewise: AssociateTimeSeriesToAssetProperty
	iotsitewise: BatchAssociateProjectAssets
	iotsitewise: BatchDisassociateProjectAssets
	iotsitewise: CreateAccessPolicy
	iotsitewise: CreateAsset
	iotsitewise: CreateAssetModel
	iotsitewise: CreateAssetModelCompositeModel
	iotsitewise: CreateBulkImportJob
	iotsitewise: CreateComputationModel
	iotsitewise: CreateDashboard
	iotsitewise: CreateDataset
	iotsitewise: CreateGateway
	iotsitewise: CreatePortal
	iotsitewise: CreateProject
	iotsitewise: DeleteAccessPolicy
	iotsitewise: DeleteAsset
	iotsitewise: DeleteAssetModel
	iotsitewise: DeleteAssetModelCompositeModel
	iotsitewise: DeleteComputationModel
	iotsitewise: DeleteDashboard

Awalan layanan	Tindakan
	iotsitewise: DeleteDataset
	iotsitewise: DeleteGateway
	iotsitewise: DeletePortal
	iotsitewise: DeleteProject
	iotsitewise: DeleteTimeSeries
	iotsitewise: DescribeAccessPolicy
	iotsitewise: DescribeAsset
	iotsitewise: DescribeAssetCompositeModel
	iotsitewise: DescribeAssetModel
	iotsitewise: DescribeAssetModelCompositeModel
	iotsitewise: DescribeAssetModelInterfaceRelationship
	iotsitewise: DescribeAssetProperty
	iotsitewise: DescribeBulkImportJob
	iotsitewise: DescribeComputationModel
	iotsitewise: DescribeComputationModelExecutionSummary
	iotsitewise: DescribeDashboard
	iotsitewise: DescribeDataset
	iotsitewise: DescribeDefaultEncryptionConfiguration
	iotsitewise: DescribeExecution
	iotsitewise: DescribeGateway
	iotsitewise: DescribeGatewayCapabilityConfiguration

Awalan layanan	Tindakan
	iotsitewise: DescribeLoggingOptions
	iotsitewise: DescribePortal
	iotsitewise: DescribeProject
	iotsitewise: DescribeStorageConfiguration
	iotsitewise: DescribeTimeSeries
	iotsitewise: DisassociateAssets
	iotsitewise: DisassociateTimeSeriesFromAssetProperty
	iotsitewise: ExecuteAction
	iotsitewise: ExecuteQuery
	iotsitewise: ListAccessPolicies
	iotsitewise: ListActions
	iotsitewise: ListAssetModelCompositeModels
	iotsitewise: ListAssetModelProperties
	iotsitewise: ListAssetModels
	iotsitewise: ListAssetProperties
	iotsitewise: ListAssetRelationships
	iotsitewise: ListAssets
	iotsitewise: ListAssociatedAssets
	iotsitewise: ListBulkImportJobs
	iotsitewise: ListCompositionRelationships
	iotsitewise: ListComputationModelDataBindingUsages

Awalan layanan	Tindakan
	iotsitewise: ListComputationModelResolveToResources
	iotsitewise: ListComputationModels
	iotsitewise: ListDashboards
	iotsitewise: ListDatasets
	iotsitewise: ListExecutions
	iotsitewise: ListGateways
	iotsitewise: ListInterfaceRelationships
	iotsitewise: ListPortals
	iotsitewise: ListProjectAssets
	iotsitewise: ListProjects
	iotsitewise: ListTimeSeries
	iotsitewise: PutDefaultEncryptionConfiguration
	iotsitewise: PutLoggingOptions
	iotsitewise: PutStorageConfiguration
	iotsitewise: UpdateAccessPolicy
	iotsitewise: UpdateAsset
	iotsitewise: UpdateAssetModel
	iotsitewise: UpdateAssetModelCompositeModel
	iotsitewise: UpdateAssetProperty
	iotsitewise: UpdateComputationModel
	iotsitewise: UpdateDashboard

Awalan layanan	Tindakan
	iotsitewise: UpdateDataset
	iotsitewise: UpdateGateway
	iotsitewise: UpdateGatewayCapabilityConfiguration
	iotsitewise: UpdatePortal
	iotsitewise: UpdateProject

Awalan layanan	Tindakan
pembuat iottwinmaker	pembuat iottwinmaker: CancelMetadataTransferJob pembuat iottwinmaker: CreateComponentType pembuat iottwinmaker: CreateEntity pembuat iottwinmaker: CreateMetadataTransferJob pembuat iottwinmaker: CreateScene pembuat iottwinmaker: CreateSyncJob pembuat iottwinmaker: CreateWorkspace pembuat iottwinmaker: DeleteComponentType pembuat iottwinmaker: DeleteEntity pembuat iottwinmaker: DeleteScene pembuat iottwinmaker: DeleteSyncJob pembuat iottwinmaker: DeleteWorkspace pembuat iottwinmaker: ExecuteQuery pembuat iottwinmaker: GetMetadataTransferJob pembuat iottwinmaker: GetPricingPlan pembuat iottwinmaker: GetScene pembuat iottwinmaker: GetSyncJob pembuat iottwinmaker: ListComponentTypes pembuat iottwinmaker: ListComponents pembuat iottwinmaker: ListEntities pembuat iottwinmaker: ListMetadataTransferJobs

Awalan layanan	Tindakan
	pembuat iottwinmaker: ListProperties pembuat iottwinmaker: ListScenes pembuat iottwinmaker: ListSyncJobs pembuat iottwinmaker: ListSyncResources pembuat iottwinmaker: ListWorkspaces pembuat iottwinmaker: UpdateComponentType pembuat iottwinmaker: UpdateEntity pembuat iottwinmaker: UpdatePricingPlan pembuat iottwinmaker: UpdateScene pembuat iottwinmaker: UpdateWorkspace

Awalan layanan	Tindakan
iotwireless	iotwireless: AssociateAwsAccountWithPartnerAccount
	iotwireless: AssociateMulticastGroupWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithFuotaTask
	iotwireless: AssociateWirelessDeviceWithMulticastGroup
	iotwireless: AssociateWirelessDeviceWithThing
	iotwireless: AssociateWirelessGatewayWithCertificate
	iotwireless: AssociateWirelessGatewayWithThing
	iotwireless: CancelMulticastGroupSession
	iotwireless: CreateDestination
	iotwireless: CreateDeviceProfile
	iotwireless: CreateFuotaTask
	iotwireless: CreateMulticastGroup
	iotwireless: CreateNetworkAnalyzerConfiguration
	iotwireless: CreateServiceProfile
	iotwireless: CreateWirelessDevice
	iotwireless: CreateWirelessGateway
	iotwireless: CreateWirelessGatewayTask
	iotwireless: CreateWirelessGatewayTaskDefinition
	iotwireless: DeleteDestination
	iotwireless: DeleteDeviceProfile
	iotwireless: DeleteFuotaTask

Awalan layanan	Tindakan
	iotwireless: DeleteMulticastGroup
	iotwireless: DeleteNetworkAnalyzerConfiguration
	iotwireless: DeleteQueuedMessages
	iotwireless: DeleteServiceProfile
	iotwireless: DeleteWirelessDevice
	iotwireless: DeleteWirelessDeviceImportTask
	iotwireless: DeleteWirelessGateway
	iotwireless: DeleteWirelessGatewayTask
	iotwireless: DeleteWirelessGatewayTaskDefinition
	iotwireless: DeregisterWirelessDevice
	iotwireless: DisassociateAwsAccountFromPartnerAccount
	iotwireless: DisassociateMulticastGroupFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromFuotaTask
	iotwireless: DisassociateWirelessDeviceFromMulticastGroup
	iotwireless: DisassociateWirelessDeviceFromThing
	iotwireless: DisassociateWirelessGatewayFromCertificate
	iotwireless: DisassociateWirelessGatewayFromThing
	iotwireless: GetDestination
	iotwireless: GetDeviceProfile
	iotwireless: GetEventConfigurationByResourceTypes
	iotwireless: GetFuotaTask

Awalan layanan	Tindakan
	iotwireless: GetLogLevelsByResourceTypes
	iotwireless: GetMetricConfiguration
	iotwireless: GetMetrics
	iotwireless: GetMulticastGroup
	iotwireless: GetMulticastGroupSession
	iotwireless: GetNetworkAnalyzerConfiguration
	iotwireless: GetPartnerAccount
	iotwireless: GetPosition
	iotwireless: GetPositionConfiguration
	iotwireless: GetPositionEstimate
	iotwireless: GetResourceEventConfiguration
	iotwireless: GetResourceLogLevel
	iotwireless: GetResourcePosition
	iotwireless: GetServiceEndpoint
	iotwireless: GetServiceProfile
	iotwireless: GetWirelessDevice
	iotwireless: GetWirelessDeviceImportTask
	iotwireless: GetWirelessDeviceStatistics
	iotwireless: GetWirelessGateway
	iotwireless: GetWirelessGatewayCertificate
	iotwireless: GetWirelessGatewayFirmwareInformation

Awalan layanan	Tindakan
	iotwireless: GetWirelessGatewayStatistics
	iotwireless: GetWirelessGatewayTask
	iotwireless: GetWirelessGatewayTaskDefinition
	iotwireless: ListDestinations
	iotwireless: ListDeviceProfiles
	iotwireless: ListDevicesForWirelessDeviceImportTask
	iotwireless: ListEventConfigurations
	iotwireless: ListFuotaTasks
	iotwireless: ListMulticastGroups
	iotwireless: ListMulticastGroupsByFuotaTask
	iotwireless: ListNetworkAnalyzerConfigurations
	iotwireless: ListPartnerAccounts
	iotwireless: ListPositionConfigurations
	iotwireless: ListQueuedMessages
	iotwireless: ListServiceProfiles
	iotwireless: ListWirelessDeviceImportTasks
	iotwireless: ListWirelessDevices
	iotwireless: ListWirelessGatewayTaskDefinitions
	iotwireless: ListWirelessGateways
	iotwireless: PutPositionConfiguration
	iotwireless: PutResourceLogLevel

Awalan layanan	Tindakan
	iotwireless: ResetAllResourceLogLevels
	iotwireless: ResetResourceLogLevel
	iotwireless: SendDataToMulticastGroup
	iotwireless: SendDataToWirelessDevice
	iotwireless: StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless: StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless: StartFuotaTask
	iotwireless: StartMulticastGroupSession
	iotwireless: StartNetworkAnalyzerStream
	iotwireless: StartSingleWirelessDeviceImportTask
	iotwireless: StartWirelessDeviceImportTask
	iotwireless: TestWirelessDevice
	iotwireless: UpdateDestination
	iotwireless: UpdateEventConfigurationByResourceTypes
	iotwireless: UpdateFuotaTask
	iotwireless: UpdateLogLevelsByResourceTypes
	iotwireless: UpdateMetricConfiguration
	iotwireless: UpdateMulticastGroup
	iotwireless: UpdateNetworkAnalyzerConfiguration
	iotwireless: UpdatePartnerAccount

Awalan layanan	Tindakan
	iotwireless: UpdatePosition
	iotwireless: UpdateResourceEventConfiguration
	iotwireless: UpdateResourcePosition
	iotwireless: UpdateWirelessDevice
	iotwireless: UpdateWirelessDeviceImportTask
	iotwireless: UpdateWirelessGateway

Awalan layanan	Tindakan
ivs	ivs: BatchGetChannel ivs: BatchGetStreamKey ivs: BatchStartViewerSessionRevocation ivs: CreateAdConfiguration ivs: CreateChannel ivs: CreateEncoderConfiguration ivs: CreateIngestConfiguration ivs: CreateParticipantToken ivs: CreatePlaybackRestrictionPolicy ivs: CreateRecordingConfiguration ivs: CreateStorageConfiguration ivs: CreateStreamKey ivs: DeleteAdConfiguration ivs: DeleteChannel ivs: DeleteEncoderConfiguration ivs: DeleteIngestConfiguration ivs: DeletePlaybackKeyPair ivs: DeletePlaybackRestrictionPolicy ivs: DeletePublicKey ivs: DeleteRecordingConfiguration ivs: DeleteStorageConfiguration

Awalan layanan	Tindakan
	ivs: DeleteStreamKey ivs: DisconnectParticipant ivs: GetAdConfiguration ivs: GetChannel ivs: GetComposition ivs: GetEncoderConfiguration ivs: GetIngestConfiguration ivs: GetParticipant ivs: GetPlaybackKeyPair ivs: GetPlaybackRestrictionPolicy ivs: GetPublicKey ivs: GetRecordingConfiguration ivs: GetStorageConfiguration ivs: GetStream ivs: GetStreamKey ivs: GetStreamSession ivs: ImportPlaybackKeyPair ivs: ImportPublicKey ivs: InsertAdBreak ivs: ListAdConfigurations ivs: ListChannels

Awalan layanan	Tindakan
	ivs: ListCompositions ivs: ListEncoderConfigurations ivs: ListIngestConfigurations ivs: ListParticipantEvents ivs: ListParticipantReplicas ivs: ListParticipants ivs: ListPlaybackKeyPairs ivs: ListPlaybackRestrictionPolicies ivs: ListPublicKeys ivs: ListRecordingConfigurations ivs: ListStorageConfigurations ivs: ListStreamKeys ivs: ListStreamSessions ivs: ListStreams ivs: PutMetadata ivs: StartComposition ivs: StartViewerSessionRevocation ivs: StopComposition ivs: StopStream ivs: UpdateAdConfiguration ivs: UpdateChannel

Awalan layanan	Tindakan
	ivs: UpdateIngestConfiguration ivs: UpdatePlaybackRestrictionPolicy
ivschat	ivschat: CreateChatToken ivschat: CreateLoggingConfiguration ivschat: CreateRoom ivschat: DeleteLoggingConfiguration ivschat: DeleteMessage ivschat: DeleteRoom ivschat: DisconnectUser ivschat: GetLoggingConfiguration ivschat: GetRoom ivschat: ListLoggingConfigurations ivschat: ListRooms ivschat: SendEvent ivschat: UpdateLoggingConfiguration ivschat: UpdateRoom

Awalan layanan	Tindakan
kafka	kafka: BatchAssociateScramSecret kafka: BatchDisassociateScramSecret kafka: CreateCluster kafka: CreateCluster V2 kafka: CreateConfiguration kafka: CreateReplicator kafka: CreateVpcConnection kafka: DeleteCluster kafka: DeleteClusterPolicy kafka: DeleteConfiguration kafka: DeleteReplicator kafka: DeleteVpcConnection kafka: DescribeCluster kafka: DescribeClusterOperation kafka: DescribeClusterOperationV2 kafka: DescribeCluster V2 kafka: DescribeConfiguration kafka: DescribeConfigurationRevision kafka: DescribeVpcConnection kafka: GetBootstrapBrokers kafka: GetClusterPolicy

Awalan layanan	Tindakan
	kafka: GetCompatibleKafkaVersions kafka: ListClientVpcConnections kafka: ListClusterOperations kafka: ListClusterOperationsV2 kafka: ListClusters kafka: ListClusters V2 kafka: ListConfigurationRevisions kafka: ListConfigurations kafka: ListKafkaVersions kafka: ListNodes kafka: ListReplicators kafka: ListScramSecrets kafka: ListVpcConnections kafka: PutClusterPolicy kafka: RebootBroker kafka: RejectClientVpcConnection kafka: UpdateBrokerCount kafka: UpdateBrokerStorage kafka: UpdateBrokerType kafka: UpdateClusterConfiguration kafka: UpdateClusterKafkaVersion

Awalan layanan	Tindakan
	kafka: UpdateConfiguration kafka: UpdateConnectivity kafka: UpdateMonitoring kafka: UpdateRebalancing kafka: UpdateReplicationInfo kafka: UpdateSecurity kafka: UpdateStorage
kafkaconnect	kafkaconnect: CreateConnector kafkaconnect: CreateCustomPlugin kafkaconnect: CreateWorkerConfiguration kafkaconnect: DeleteConnector kafkaconnect: DeleteCustomPlugin kafkaconnect: DeleteWorkerConfiguration kafkaconnect: DescribeConnector kafkaconnect: DescribeCustomPlugin kafkaconnect: DescribeWorkerConfiguration kafkaconnect: ListConnectorOperations kafkaconnect: ListConnectors kafkaconnect: ListCustomPlugins kafkaconnect: ListWorkerConfigurations kafkaconnect: UpdateConnector

Awalan layanan	Tindakan
kendra	kendra: AssociateEntitiesToExperience kendra: AssociatePersonasToEntities kendra: BatchDeleteDocument kendra: BatchDeleteFeaturedResultsSet kendra: BatchGetDocumentStatus kendra: BatchPutDocument kendra: ClearQuerySuggestions kendra: CreateAccessControlConfiguration kendra: CreateDataSource kendra: CreateExperience kendra: CreateFaq kendra: CreateFeaturedResultsSet kendra: CreateIndex kendra: CreateQuerySuggestionsBlockList kendra: CreateThesaurus kendra: DeleteDataSource kendra: DeleteExperience kendra: DeleteFaq kendra: DeleteIndex kendra: DeletePrincipalMapping kendra: DeleteQuerySuggestionsBlockList

Awalan layanan	Tindakan
	kendra: DeleteThesaurus kendra: DescribeAccessControlConfiguration kendra: DescribeDataSource kendra: DescribeExperience kendra: DescribeFaq kendra: DescribeFeaturedResultsSet kendra: DescribeIndex kendra: DescribePrincipalMapping kendra: DescribeQuerySuggestionsBlockList kendra: DescribeQuerySuggestionsConfig kendra: DescribeThesaurus kendra: DisassociateEntitiesFromExperience kendra: DisassociatePersonasFromEntities kendra: GetQuerySuggestions kendra: GetSnapshots kendra: ListAccessControlConfigurations kendra: ListDataSourceSyncJobs kendra: ListDataSources kendra: ListEntityPersonas kendra: ListExperienceEntities kendra: ListExperiences

Awalan layanan	Tindakan
	kendra: ListFaqs kendra: ListFeaturedResultsSets kendra: ListGroupsOlderThanOrderingId kendra: ListIndices kendra: ListQuerySuggestionsBlockLists kendra: ListThesauri kendra: PutPrincipalMapping Kendra:Pertanyaan Kendra:ambil kendra: StartDataSourceSyncJob kendra: StopDataSourceSyncJob kendra: SubmitFeedback kendra: UpdateDataSource kendra: UpdateExperience kendra: UpdateFeaturedResultsSet kendra: UpdateIndex kendra: UpdateQuerySuggestionsBlockList kendra: UpdateQuerySuggestionsConfig kendra: UpdateThesaurus

Awalan layanan	Tindakan
kinesis	kinesis: CreateStream
	kinesis: DecreaseStreamRetentionPeriod
	kinesis: DeleteStream
	kinesis: DeregisterStreamConsumer
	kinesis: DescribeAccountSettings
	kinesis: DescribeLimits
	kinesis: DescribeStream
	kinesis: DescribeStreamConsumer
	kinesis: DescribeStreamSummary
	kinesis: DisableEnhancedMonitoring
	kinesis: EnableEnhancedMonitoring
	kinesis: IncreaseStreamRetentionPeriod
	kinesis: ListShards
	kinesis: ListStreamConsumers
	kinesis: ListStreams
	kinesis: MergeShards
	kinesis: RegisterStreamConsumer
	kinesis: SplitShard
	kinesis: StartStreamEncryption
	kinesis: StopStreamEncryption
	kinesis: UpdateAccountSettings

Awalan layanan	Tindakan
	kinesis: UpdateShardCount kinesis: UpdateStreamMode

Awalan layanan	Tindakan
kinesisanalitik	kinesisanalitik: AddApplicationCloudWatchLoggingOption kinesisanalitik: AddApplicationInput kinesisanalitik: AddApplicationInputProcessingConfiguration kinesisanalitik: AddApplicationOutput kinesisanalitik: AddApplicationReferenceDataSource kinesisanalitik: AddApplicationVpcConfiguration kinesisanalitik: CreateApplication kinesisanalitik: CreateApplicationPresignedUrl kinesisanalitik: CreateApplicationSnapshot kinesisanalitik: DeleteApplication kinesisanalitik: DeleteApplicationCloudWatchLoggingOption kinesisanalitik: DeleteApplicationInputProcessingConfiguration kinesisanalitik: DeleteApplicationOutput kinesisanalitik: DeleteApplicationReferenceDataSource kinesisanalitik: DeleteApplicationSnapshot kinesisanalitik: DeleteApplicationVpcConfiguration kinesisanalitik: DescribeApplication kinesisanalitik: DescribeApplicationOperation kinesisanalitik: DescribeApplicationSnapshot kinesisanalitik: DescribeApplicationVersion kinesisanalitik: DiscoverInputSchema

Awalan layanan	Tindakan
	kinesisanalitik: ListApplicationOperations kinesisanalitik: ListApplicationSnapshots kinesisanalitik: ListApplicationVersions kinesisanalitik: ListApplications kinesisanalitik: RollbackApplication kinesisanalitik: StartApplication kinesisanalitik: StopApplication kinesisanalitik: UpdateApplication kinesisanalitik: UpdateApplicationMaintenanceConfiguration

Awalan layanan	Tindakan
km	km: CancelKeyDeletion km: ConnectCustomKeyStore km: CreateAlias km: CreateCustomKeyStore km: CreateGrant km: CreateKey kms:Decrypt km: DeleteAlias km: DeleteCustomKeyStore km: DeleteImportedKeyMaterial km: DeriveSharedSecret km: DescribeCustomKeyStores km: DescribeKey km: DisableKey km: DisableKeyRotation km: DisconnectCustomKeyStore km: EnableKey km: EnableKeyRotation kms:Encrypt km: GenerateDataKey km: GenerateDataKeyPair

Awalan layanan	Tindakan
	km: GenerateDataKeyPairWithoutPlaintext
	km: GenerateDataKeyWithoutPlaintext
	km: GenerateMac
	km: GenerateRandom
	km: GetKeyLastUsage
	km: GetKeyPolicy
	km: GetKeyRotationStatus
	km: GetParametersForImport
	km: GetPublicKey
	km: ImportKeyMaterial
	km: ListAliases
	km: ListGrants
	km: ListKeyPolicies
	km: ListKeyRotations
	km: ListKeys
	km: ListRetirableGrants
	km: ReplicateKey
	km: RetireGrant
	km: RevokeGrant
	km: RotateKeyOnDemand
	km: ScheduleKeyDeletion

Awalan layanan	Tindakan
	KMS: Tanda
	km: UpdateAlias
	km: UpdateCustomKeyStore
	km: UpdateKeyDescription
	km: UpdatePrimaryRegion
	KMS: Verifikasi
	km: VerifyMac

Awalan layanan	Tindakan
lambda	lambda: AddLayerVersionPermission lambda: AddPermission lambda: CreateAlias lambda: CreateCodeSigningConfig lambda: CreateEventSourceMapping lambda: CreateFunction lambda: CreateFunctionUrlConfig lambda: DeleteAlias lambda: DeleteCapacityProvider lambda: DeleteCodeSigningConfig lambda: DeleteEventSourceMapping lambda: DeleteFunction lambda: DeleteFunctionCodeSigningConfig lambda: DeleteFunctionConcurrency lambda: DeleteFunctionEventInvokeConfig lambda: DeleteFunctionUrlConfig lambda: DeleteLayerVersion lambda: DeleteProvisionedConcurrencyConfig lambda: GetAccountSettings lambda: GetAlias lambda: GetCapacityProvider

Awalan layanan	Tindakan
	lambda: GetCodeSigningConfig lambda: GetEventSourceMapping lambda: GetFunction lambda: GetFunctionCodeSigningConfig lambda: GetFunctionConcurrency lambda: GetFunctionConfiguration lambda: GetFunctionEventInvokeConfig lambda: GetFunctionRecursionConfig lambda: GetFunctionScalingConfig lambda: GetFunctionUrlConfig lambda: GetLayerVersion lambda: GetLayerVersionPolicy lambda: GetPolicy lambda: GetProvisionedConcurrencyConfig lambda: GetRuntimeManagementConfig lambda: ListAliases lambda: ListCapacityProviders lambda: ListCodeSigningConfigs lambda: ListDurableExecutionsByFunction lambda: ListEventSourceMappings lambda: ListFunctionEventInvokeConfigs

Awalan layanan	Tindakan
	lambda: ListFunctionUrlConfigs lambda: ListFunctions lambda: ListFunctionsByCodeSigningConfig lambda: ListLayerVersions lambda: ListLayers lambda: ListProvisionedConcurrencyConfigs lambda: ListVersionsByFunction lambda: PublishLayerVersion lambda: PublishVersion lambda: PutFunctionCodeSigningConfig lambda: PutFunctionConcurrency lambda: PutFunctionEventInvokeConfig lambda: PutFunctionRecursionConfig lambda: PutFunctionScalingConfig lambda: PutProvisionedConcurrencyConfig lambda: PutRuntimeManagementConfig lambda: RemoveLayerVersionPermission lambda: RemovePermission lambda: UpdateAlias lambda: UpdateCapacityProvider lambda: UpdateCodeSigningConfig

Awalan layanan	Tindakan
	lambda: UpdateEventSourceMapping lambda: UpdateFunctionCode lambda: UpdateFunctionConfiguration lambda: UpdateFunctionEventInvokeConfig lambda: UpdateFunctionUrlConfig

Awalan layanan	Tindakan
lex	lex: BatchCreateCustomVocabularyItem
	lex: BatchDeleteCustomVocabularyItem
	lex: BatchUpdateCustomVocabularyItem
	lex: BuildBotLocale
	lex: CreateBotAlias
	lex: CreateBotReplica
	lex: CreateBotVersion
	lex: CreateExport
	lex: CreateIntentVersion
	lex: CreateResourcePolicy
	lex: CreateSlotTypeVersion
	lex: CreateTestSetDiscrepancyReport
	lex: CreateUploadUrl
	lex: DeleteBot
	lex: DeleteBotAnalyzerRecommendation
	lex: DeleteBotChannelAssociation
	lex: DeleteBotReplica
	lex: DeleteExport
	lex: DeleteImport
	lex: DeleteIntentVersion
	lex: DeleteResourcePolicy

Awalan layanan	Tindakan
	lex: DeleteSlotTypeVersion
	lex: DeleteTestSet
	lex: DeleteUtterances
	lex: DescribeBotAlias
	lex: DescribeBotAnalyzerRecommendation
	lex: DescribeBotRecommendation
	lex: DescribeBotReplica
	lex: DescribeBotResourceGeneration
	lex: DescribeBotVersion
	lex: DescribeCustomVocabularyMetadata
	lex: DescribeExport
	lex: DescribeImport
	lex: DescribeResourcePolicy
	lex: DescribeTestExecution
	lex: DescribeTestSet
	lex: DescribeTestSetDiscrepancyReport
	lex: DescribeTestSetGeneration
	lex: GenerateBotElement
	lex: GetBot
	lex: GetBotAlias
	lex: GetBotAliases

Awalan layanan	Tindakan
	lex: GetBotChannelAssociation
	lex: GetBotChannelAssociations
	lex: GetBotVersions
	lex: GetBots
	lex: GetBuiltinIntent
	lex: GetBuiltinIntents
	lex: GetBuiltinSlotTypes
	lex: GetExport
	lex: GetImport
	lex: GetIntent
	lex: GetIntentVersions
	lex: GetIntents
	lex: GetMigration
	lex: GetMigrations
	lex: GetSlotType
	lex: GetSlotTypeVersions
	lex: GetSlotTypes
	lex: GetTestExecutionArtifactsUrl
	lex: GetUtterancesView
	lex: ListBotAliasReplicas
	lex: ListBotAliases

Awalan layanan	Tindakan
	lex: ListBotRecommendations
	lex: ListBotReplicas
	lex: ListBotResourceGenerations
	lex: ListBotVersionReplicas
	lex: ListBotVersions
	lex: ListBots
	lex: ListBuiltInIntents
	lex: ListBuiltInSlotTypes
	lex: ListCustomVocabularyItems
	lex: ListExports
	lex: ListImports
	lex: ListIntentMetrics
	lex: ListIntentPaths
	lex: ListRecommendedIntents
	lex: ListSessionAnalyticsData
	lex: ListSessionMetrics
	lex: ListTestExecutionResultItems
	lex: ListTestExecutions
	lex: ListTestSets
	lex: PutBot
	lex: PutBotAlias

Awalan layanan	Tindakan
	lex: PutIntent
	lex: PutSlotType
	lex: SearchAssociatedTranscripts
	lex: StartBotAnalyzer
	lex: StartBotRecommendation
	lex: StartImport
	lex: StartMigration
	lex: StartTestExecution
	lex: StartTestSetGeneration
	lex: StopBotAnalyzer
	lex: StopBotRecommendation
	lex: UpdateBotAlias
	lex: UpdateBotRecommendation
	lex: UpdateExport
	lex: UpdateResourcePolicy

Awalan layanan	Tindakan
lisensi-manajer-linux-langganan	lisensi-manajer-linux-langganan: DeregisterSubscriptionProvider lisensi-manajer-linux-langganan: GetRegisteredSubscriptionProvider lisensi-manajer-linux-langganan: GetServiceSettings lisensi-manajer-linux-langganan: ListLinuxSubscriptionInstances lisensi-manajer-linux-langganan: ListLinuxSubscriptions lisensi-manajer-linux-langganan: ListRegisteredSubscriptionProviders lisensi-manajer-linux-langganan: RegisterSubscriptionProvider lisensi-manajer-linux-langganan: UpdateServiceSettings

Awalan layanan	Tindakan
lightsail	lightsail: AllocateStaticIp
	lightsail: AttachCertificateToDistribution
	lightsail: AttachDisk
	lightsail: AttachInstancesToLoadBalancer
	lightsail: AttachLoadBalancerTlsCertificate
	lightsail: AttachStaticIp
	lightsail: CloseInstancePublicPorts
	lightsail: CopySnapshot
	lightsail: CreateBucket
	lightsail: CreateBucketAccessKey
	lightsail: CreateCertificate
	lightsail: CreateCloudFormationStack
	lightsail: CreateContactMethod
	lightsail: CreateContainerService
	lightsail: CreateContainerServiceDeployment
	lightsail: CreateContainerServiceRegistryLogin
	lightsail: CreateDisk
	lightsail: CreateDiskFromSnapshot
	lightsail: CreateDiskSnapshot
	lightsail: CreateDistribution
	lightsail: CreateDomain

Awalan layanan	Tindakan
	lightsail: CreateGUISessionAccessDetails
	lightsail: CreateInstanceSnapshot
	lightsail: CreateInstances
	lightsail: CreateInstancesFromSnapshot
	lightsail: CreateKeyPair
	lightsail: CreateLoadBalancer
	lightsail: CreateLoadBalancerTlsCertificate
	lightsail: CreateRelationalDatabase
	lightsail: CreateRelationalDatabaseFromSnapshot
	lightsail: CreateRelationalDatabaseSnapshot
	lightsail: DeleteAlarm
	lightsail: DeleteAutoSnapshot
	lightsail: DeleteBucket
	lightsail: DeleteBucketAccessKey
	lightsail: DeleteCertificate
	lightsail: DeleteContactMethod
	lightsail: DeleteContainerImage
	lightsail: DeleteContainerService
	lightsail: DeleteDisk
	lightsail: DeleteDiskSnapshot
	lightsail: DeleteDistribution

Awalan layanan	Tindakan
	lightsail: DeleteDomain
	lightsail: DeleteDomainEntry
	lightsail: DeleteInstance
	lightsail: DeleteInstanceSnapshot
	lightsail: DeleteKeyPair
	lightsail: DeleteKnownHostKeys
	lightsail: DeleteLoadBalancer
	lightsail: DeleteLoadBalancerTlsCertificate
	lightsail: DeleteRelationalDatabase
	lightsail: DeleteRelationalDatabaseSnapshot
	lightsail: DetachCertificateFromDistribution
	lightsail: DetachDisk
	lightsail: DetachInstancesFromLoadBalancer
	lightsail: DetachStaticIp
	lightsail: DisableAddOn
	lightsail: DownloadDefaultKeyPair
	lightsail: EnableAddOn
	lightsail: ExportSnapshot
	lightsail: GetActiveNames
	lightsail: GetAlarms
	lightsail: GetAutoSnapshots

Awalan layanan	Tindakan
	lightsail: GetBlueprints
	lightsail: GetBucketAccessKeys
	lightsail: GetBucketBundles
	lightsail: GetBucketMetricData
	lightsail: GetBuckets
	lightsail: GetBundles
	lightsail: GetCertificates
	lightsail: GetCloudFormationStackRecords
	lightsail: GetContactMethods
	lightsail: GetContainerAPIMetadata
	lightsail: GetContainerImages
	lightsail: GetContainerLog
	lightsail: GetContainerServiceDeployments
	lightsail: GetContainerServiceMetricData
	lightsail: GetContainerServicePowers
	lightsail: GetContainerServices
	lightsail: GetCostEstimate
	lightsail: GetDisk
	lightsail: GetDiskSnapshot
	lightsail: GetDiskSnapshots
	lightsail: GetDisks

Awalan layanan	Tindakan
	lightsail: GetDistributionBundles
	lightsail: GetDistributionLatestCacheReset
	lightsail: GetDistributionMetricData
	lightsail: GetDistributions
	lightsail: GetDomain
	lightsail: GetExportSnapshotRecords
	lightsail: GetInstance
	lightsail: GetInstanceMetricData
	lightsail: GetInstancePortStates
	lightsail: GetInstanceSnapshot
	lightsail: GetInstanceSnapshots
	lightsail: GetInstanceState
	lightsail: GetInstances
	lightsail: GetKeyPair
	lightsail: GetKeyPairs
	lightsail: GetLoadBalancer
	lightsail: GetLoadBalancerMetricData
	lightsail: GetLoadBalancerTlsCertificates
	lightsail: GetLoadBalancerTlsPolicies
	lightsail: GetLoadBalancers
	lightsail: GetOperation

Awalan layanan	Tindakan
	lightsail: GetOperations
	lightsail: GetOperationsForResource
	lightsail: GetRegions
	lightsail: GetRelationalDatabase
	lightsail: GetRelationalDatabaseBlueprints
	lightsail: GetRelationalDatabaseBundles
	lightsail: GetRelationalDatabaseEvents
	lightsail: GetRelationalDatabaseLogEvents
	lightsail: GetRelationalDatabaseLogStreams
	lightsail: GetRelationalDatabaseMasterUserPassword
	lightsail: GetRelationalDatabaseMetricData
	lightsail: GetRelationalDatabaseParameters
	lightsail: GetRelationalDatabaseSnapshot
	lightsail: GetRelationalDatabaseSnapshots
	lightsail: GetRelationalDatabases
	lightsail: GetSetupHistory
	lightsail: GetStaticIp
	lightsail: GetStaticIps
	lightsail: ImportKeyPair
	lightsail: IsVpcPeered
	lightsail: OpenInstancePublicPorts

Awalan layanan	Tindakan
	lightsail: PeerVpc
	lightsail: PutAlarm
	lightsail: PutInstancePublicPorts
	lightsail: RebootInstance
	lightsail: RebootRelationalDatabase
	lightsail: RegisterContainerImage
	lightsail: ReleaseStaticIp
	lightsail: ResetDistributionCache
	lightsail: SendContactMethodVerification
	lightsail: SetIpAddressType
	lightsail: SetResourceAccessForBucket
	lightsail: SetupInstanceHttps
	LightSail: startGuiSession
	lightsail: StartInstance
	lightsail: StartRelationalDatabase
	LightSail: StopGuiSession
	lightsail: StopInstance
	lightsail: StopRelationalDatabase
	lightsail: TestAlarm
	lightsail: UnpeerVpc
	lightsail: UpdateBucket

Awalan layanan	Tindakan
	lightsail: UpdateBucketBundle
	lightsail: UpdateContainerService
	lightsail: UpdateDistribution
	lightsail: UpdateDistributionBundle
	lightsail: UpdateDomainEntry
	lightsail: UpdateInstanceMetadataOptions
	lightsail: UpdateLoadBalancerAttribute
	lightsail: UpdateRelationalDatabase
	lightsail: UpdateRelationalDatabaseParameters

Awalan layanan	Tindakan
log	log: AssociateKmsKey log: AssociateSourceToS3TableIntegration log: CancelExportTask log: CancelImportTask log: CreateDelivery log: CreateExportTask log: CreateLogAnomalyDetector log: CreateLogGroup log: CreateLogStream log: CreateLookupTable log: DeleteDataProtectionPolicy log: DeleteDelivery log: DeleteDeliveryDestination log: DeleteDeliveryDestinationPolicy log: DeleteDeliverySource log: DeleteDestination log: DeleteIndexPolicy log: DeleteIntegration log: DeleteLogAnomalyDetector log: DeleteLogGroup log: DeleteLogStream

Awalan layanan	Tindakan
	log: DeleteLookupTable
	log: DeleteMetricFilter
	log: DeleteQueryDefinition
	log: DeleteResourcePolicy
	log: DeleteRetentionPolicy
	log: DeleteScheduledQuery
	log: DeleteSubscriptionFilter
	log: DeleteSyslogConfiguration
	log: DeleteTransformer
	log: DescribeAccountPolicies
	log: DescribeConfigurationTemplates
	log: DescribeDeliveries
	log: DescribeDeliveryDestinations
	log: DescribeDeliverySources
	log: DescribeDestinations
	log: DescribeExportTasks
	log: DescribeFieldIndexes
	log: DescribeImportTaskBatches
	log: DescribeImportTasks
	log: DescribeIndexPolicies
	log: DescribeLogGroups

Awalan layanan	Tindakan
	log: DescribeLogStreams log: DescribeLookupTables log: DescribeMetricFilters log: DescribeQueries log: DescribeQueryDefinitions log: DescribeResourcePolicies log: DescribeSubscriptionFilters log: DisassociateKmsKey log: DisassociateSourceFromS3TableIntegration log: GetDataProtectionPolicy log: GetDelivery log: GetDeliveryDestination log: GetDeliveryDestinationPolicy log: GetDeliverySource log: GetIntegration log: GetLogAnomalyDetector log: GetLogFields log: GetLogGroupFields log: GetLogRecord log: GetLookupTable log: GetQueryResults

Awalan layanan	Tindakan
	log: GetScheduledQuery log: GetScheduledQueryHistory log: GetStorageTierPolicy log: GetTransformer log: ListAnomalies log: ListIntegrations log: ListLogAnomalyDetectors log: ListLogGroupsForQuery log: ListScheduledQueries log: ListSourcesForS3TableIntegration log: ListSyslogConfigurations log: PutBearerTokenAuthentication log: PutDataProtectionPolicy log: PutDeliveryDestination log: PutDeliveryDestinationPolicy log: PutDeliverySource log: PutDestination log: PutDestinationPolicy log: PutIndexPolicy log: PutIntegration log: PutLogGroupDeletionProtection

Awalan layanan	Tindakan
	log: PutMetricFilter log: PutQueryDefinition log: PutResourcePolicy log: PutRetentionPolicy log: PutStorageTierPolicy log: PutSubscriptionFilter log: PutSyslogConfiguration log: PutTransformer log: StartLiveTail log: StartQuery log: StopQuery log: TestMetricFilter log: TestTransformer log: UpdateAnomaly log: UpdateDeliveryConfiguration log: UpdateLogAnomalyDetector log: UpdateLookupTable

Awalan layanan	Tindakan
peralatan pencarian	Lookoutequipment: CreateDataset
	Lookoutequipment: CreateInferenceScheduler
	Lookoutequipment: CreateLabel
	Lookoutequipment: CreateLabelGroup
	Lookoutequipment: CreateModel
	Lookoutequipment: DeleteDataset
	Lookoutequipment: DeleteInferenceScheduler
	Lookoutequipment: DeleteLabel
	Lookoutequipment: DeleteLabelGroup
	Lookoutequipment: DeleteModel
	Lookoutequipment: DeleteResourcePolicy
	Lookoutequipment: DeleteRetrainingScheduler
	Lookoutequipment: DescribeDataIngestionJob
	Lookoutequipment: DescribeDataset
	Lookoutequipment: DescribeInferenceScheduler
	Lookoutequipment: DescribeLabelGroup
	Lookoutequipment: DescribeModel
	Lookoutequipment: DescribeModelVersion
	Lookoutequipment: DescribeResourcePolicy
	Lookoutequipment: DescribeRetrainingScheduler
	LookouteQuipment:Describelabel

Awalan layanan	Tindakan
	Lookoutequipment: ImportDataset
	Lookoutequipment: ImportModelVersion
	Lookoutequipment: ListDataIngestionJobs
	Lookoutequipment: ListDatasets
	Lookoutequipment: ListInferenceEvents
	Lookoutequipment: ListInferenceExecutions
	Lookoutequipment: ListInferenceSchedulers
	Lookoutequipment: ListLabelGroups
	Lookoutequipment: ListLabels
	Lookoutequipment: ListModelVersions
	Lookoutequipment: ListModels
	Lookoutequipment: ListRetrainingSchedulers
	Lookoutequipment: ListSensorStatistics
	Lookoutequipment: PutResourcePolicy
	Lookoutequipment: StartDataIngestionJob
	Lookoutequipment: StartInferenceScheduler
	Lookoutequipment: StartRetrainingScheduler
	Lookoutequipment: StopInferenceScheduler
	Lookoutequipment: StopRetrainingScheduler
	Lookoutequipment: UpdateActiveModelVersion
	Lookoutequipment: UpdateInferenceScheduler

Awalan layanan	Tindakan
	Lookoutequipment: UpdateLabelGroup Lookoutequipment: UpdateModel Lookoutequipment: UpdateRetrainingScheduler

Awalan layanan	Tindakan
lookoutmetrics	lookoutmetrics: ActivateAnomalyDetector
	lookoutmetrics: BackTestAnomalyDetector
	lookoutmetrics: CreateAlert
	lookoutmetrics: CreateAnomalyDetector
	lookoutmetrics: CreateMetricSet
	lookoutmetrics: DeactivateAnomalyDetector
	lookoutmetrics: DeleteAlert
	lookoutmetrics: DeleteAnomalyDetector
	lookoutmetrics: DescribeAlert
	lookoutmetrics: DescribeAnomalyDetectionExecutions
	lookoutmetrics: DescribeAnomalyDetector
	lookoutmetrics: DescribeMetricSet
	lookoutmetrics: DetectMetricSetConfig
	lookoutmetrics: GetAnomalyGroup
	lookoutmetrics: GetDataQualityMetrics
	lookoutmetrics: GetFeedback
	lookoutmetrics: GetSampleData
	lookoutmetrics: ListAlerts
	lookoutmetrics: ListAnomalyDetectors
	lookoutmetrics: ListAnomalyGroupRelatedMetrics
	lookoutmetrics: ListAnomalyGroupSummaries

Awalan layanan	Tindakan
	lookoutmetrics: ListAnomalyGroupTimeSeries lookoutmetrics: ListMetricSets lookoutmetrics: PutFeedback lookoutmetrics: UpdateAlert lookoutmetrics: UpdateAnomalyDetector lookoutmetrics: UpdateMetricSet

Awalan layanan	Tindakan
m2	m2: CancelBatchJobExecution
	m2: CreateApplication
	m2: CreateDataSetExportTask
	m2: CreateDataSetImportTask
	m2: CreateDeployment
	m2: CreateEnvironment
	m2: DeleteApplication
	m2: DeleteApplicationFromEnvironment
	m2: DeleteEnvironment
	m2: GetApplication
	m2: GetApplicationVersion
	m2: GetBatchJobExecution
	m2: GetDataSetDetails
	m2: GetDataSetExportTask
	m2: GetDataSetImportTask
	m2: GetDeployment
	m2: GetEnvironment
	m2: GetSignedBluinsightsUrl
	m2: ListApplicationVersions
	m2: ListApplications
	m2: ListBatchJobDefinitions

Awalan layanan	Tindakan
	m2: ListBatchJobExecutions
	m2: ListBatchJobRestartPoints
	m2: ListDataSetExportHistory
	m2: ListDataSetImportHistory
	m2: ListDataSets
	m2: ListDeployments
	m2: ListEngineVersions
	m2: ListEnvironments
	m2: StartApplication
	m2: StartBatchJob
	m2: StopApplication
	m2: UpdateApplication
	m2: UpdateEnvironment

Awalan layanan	Tindakan
terkelolablockchain	terkelolablockchain: CreateAccessor terkelolablockchain: CreateMember terkelolablockchain: CreateNetwork terkelolablockchain: CreateNode terkelolablockchain: CreateProposal terkelolablockchain: DeleteAccessor terkelolablockchain: DeleteMember terkelolablockchain: DeleteNode terkelolablockchain: GetAccessor terkelolablockchain: GetMember terkelolablockchain: GetNetwork terkelolablockchain: GetNode terkelolablockchain: GetProposal terkelolablockchain: InvokeRpcPolygonMainnet terkelolablockchain: InvokeRpcPolygonMumbaiTestnet terkelolablockchain: ListAccessors terkelolablockchain: ListInvitations terkelolablockchain: ListMembers terkelolablockchain: ListNetworks terkelolablockchain: ListNodes terkelolablockchain: ListProposalVotes

Awalan layanan	Tindakan
	terkelolablockchain: ListProposals terkelolablockchain: RejectInvitation terkelolablockchain: UpdateMember terkelolablockchain: UpdateNode terkelolablockchain: VoteOnProposal

Awalan layanan	Tindakan
mediaconnect	mediaconnect: AddBridgeOutputs
	mediaconnect: AddBridgeSources
	mediaconnect: AddFlowMediaStreams
	mediaconnect: AddFlowOutputs
	mediaconnect: AddFlowSources
	mediaconnect: AddFlowVpcInterfaces
	mediaconnect: CreateBridge
	mediaconnect: CreateFlow
	mediaconnect: CreateGateway
	mediaconnect: DeleteBridge
	mediaconnect: DeleteFlow
	mediaconnect: DeleteGateway
	mediaconnect: DeleteRouterInput
	mediaconnect: DeleteRouterNetworkInterface
	mediaconnect: DeleteRouterOutput
	mediaconnect: DeregisterGatewayInstance
	mediaconnect: DescribeBridge
	mediaconnect: DescribeFlow
	mediaconnect: DescribeFlowSourceMetadata
	mediaconnect: DescribeFlowSourceThumbnail
	mediaconnect: DescribeGateway

Awalan layanan	Tindakan
	mediaconnect: DescribeGatewayInstance
	mediaconnect: DescribeOffering
	mediaconnect: DescribeReservation
	mediaconnect: GetRouterInput
	mediaconnect: GetRouterInputSourceMetadata
	mediaconnect: GetRouterInputThumbnail
	mediaconnect: GetRouterNetworkInterface
	mediaconnect: GetRouterOutput
	mediaconnect: GrantFlowEntitlements
	mediaconnect: ListBridges
	mediaconnect: ListEntitlements
	mediaconnect: ListFlows
	mediaconnect: ListGatewayInstances
	mediaconnect: ListGateways
	mediaconnect: ListOfferings
	mediaconnect: ListReservations
	mediaconnect: ListRouterInputs
	mediaconnect: ListRouterNetworkInterfaces
	mediaconnect: ListRouterOutputs
	mediaconnect: PurchaseOffering
	mediaconnect: RemoveBridgeOutput

Awalan layanan	Tindakan
	mediaconnect: RemoveBridgeSource
	mediaconnect: RemoveFlowMediaStream
	mediaconnect: RemoveFlowOutput
	mediaconnect: RemoveFlowSource
	mediaconnect: RemoveFlowVpcInterface
	mediaconnect: RestartRouterInput
	mediaconnect: RestartRouterOutput
	mediaconnect: RevokeFlowEntitlement
	mediaconnect: StartFlow
	mediaconnect: StartRouterInput
	mediaconnect: StartRouterOutput
	mediaconnect: StopFlow
	mediaconnect: StopRouterInput
	mediaconnect: StopRouterOutput
	mediaconnect: TakeRouterInput
	mediaconnect: UpdateBridge
	mediaconnect: UpdateBridgeOutput
	mediaconnect: UpdateBridgeSource
	mediaconnect: UpdateBridgeState
	mediaconnect: UpdateFlow
	mediaconnect: UpdateFlowEntitlement

Awalan layanan	Tindakan
	mediaconnect: UpdateFlowMediaStream mediaconnect: UpdateGatewayInstance

Awalan layanan	Tindakan
mediaconvert	mediaconvert: AssociateCertificate mediaconvert: CancelJob mediaconvert: CreateJob mediaconvert: CreateJobTemplate mediaconvert: CreatePreset mediaconvert: CreateQueue mediaconvert: CreateResourceShare mediaconvert: DeleteJobTemplate mediaconvert: DeletePolicy mediaconvert: DeletePreset mediaconvert: DeleteQueue mediaconvert: DescribeEndpoints mediaconvert: DisassociateCertificate mediaconvert: GetJob mediaconvert: GetJobTemplate mediaconvert: GetPolicy mediaconvert: GetPreset mediaconvert: GetQueue mediaconvert: ListJobTemplates mediaconvert: ListJobs mediaconvert: ListPresets

Awalan layanan	Tindakan
	mediaconvert: ListQueues mediaconvert: ListVersions MediaConvert:Probe mediaconvert: PutPolicy mediaconvert: SearchJobs mediaconvert: UpdateJobTemplate mediaconvert: UpdatePreset mediaconvert: UpdateQueue

Awalan layanan	Tindakan
medialive	medialive: AcceptInputDeviceTransfer
	medialive: BatchDelete
	medialive: BatchStart
	medialive: BatchStop
	medialive: BatchUpdateSchedule
	medialive: CancelInputDeviceTransfer
	medialive: ClaimDevice
	medialive: CreateChannel
	medialive: CreateChannelPlacementGroup
	medialive: CreateCloudWatchAlarmTemplate
	medialive: CreateCloudWatchAlarmTemplateGroup
	medialive: CreateCluster
	medialive: CreateEventBridgeRuleTemplate
	medialive: CreateEventBridgeRuleTemplateGroup
	medialive: CreateInput
	medialive: CreateInputSecurityGroup
	medialive: CreateMultiplex
	medialive: CreateMultiplexProgram
	medialive: CreateNetwork
	medialive: CreateNode
	medialive: CreateNodeRegistrationScript

Awalan layanan	Tindakan
	medialive: CreatePartnerInput
	medialive: CreateSdiSource
	medialive: CreateSignalMap
	medialive: DeleteChannel
	medialive: DeleteChannelPlacementGroup
	medialive: DeleteCloudWatchAlarmTemplate
	medialive: DeleteCloudWatchAlarmTemplateGroup
	medialive: DeleteCluster
	medialive: DeleteEventBridgeRuleTemplate
	medialive: DeleteEventBridgeRuleTemplateGroup
	medialive: DeleteInput
	medialive: DeleteInputSecurityGroup
	medialive: DeleteMultiplex
	medialive: DeleteMultiplexProgram
	medialive: DeleteNetwork
	medialive: DeleteNode
	medialive: DeleteReservation
	medialive: DeleteSchedule
	medialive: DeleteSdiSource
	medialive: DeleteSignalMap
	medialive: DescribeAccountConfiguration

Awalan layanan	Tindakan
	medialive: DescribeChannel
	medialive: DescribeChannelPlacementGroup
	medialive: DescribeCluster
	medialive: DescribeInput
	medialive: DescribeInputDevice
	medialive: DescribeInputDeviceThumbnail
	medialive: DescribeInputSecurityGroup
	medialive: DescribeMultiplex
	medialive: DescribeMultiplexProgram
	medialive: DescribeNetwork
	medialive: DescribeNode
	medialive: DescribeOffering
	medialive: DescribeReservation
	medialive: DescribeSchedule
	medialive: DescribeSdiSource
	medialive: DescribeThumbnails
	medialive: GetCloudWatchAlarmTemplate
	medialive: GetCloudWatchAlarmTemplateGroup
	medialive: GetEventBridgeRuleTemplate
	medialive: GetEventBridgeRuleTemplateGroup
	medialive: GetSignalMap

Awalan layanan	Tindakan
	medialive: ListAlerts
	medialive: ListChannelPlacementGroups
	medialive: ListChannels
	medialive: ListCloudWatchAlarmTemplateGroups
	medialive: ListCloudWatchAlarmTemplates
	medialive: ListClusterAlerts
	medialive: ListClusters
	medialive: ListEventBridgeRuleTemplateGroups
	medialive: ListEventBridgeRuleTemplates
	medialive: ListInputDeviceTransfers
	medialive: ListInputDevices
	medialive: ListInputSecurityGroups
	medialive: ListInputs
	medialive: ListMultiplexAlerts
	medialive: ListMultiplexPrograms
	medialive: ListMultiplexes
	medialive: ListNetworks
	medialive: ListNodes
	medialive: ListOfferings
	medialive: ListReservations
	medialive: ListSdiSources

Awalan layanan	Tindakan
	medialive: ListSignalMaps
	medialive: ListVersions
	medialive: PurchaseOffering
	medialive: RebootInputDevice
	medialive: RejectInputDeviceTransfer
	medialive: RestartChannelPipelines
	medialive: StartChannel
	medialive: StartDeleteMonitorDeployment
	medialive: StartInputDevice
	medialive: StartInputDeviceMaintenanceWindow
	medialive: StartMonitorDeployment
	medialive: StartMultiplex
	medialive: StartUpdateSignalMap
	medialive: StopChannel
	medialive: StopInputDevice
	medialive: StopMultiplex
	medialive: TransferInputDevice
	medialive: UpdateAccountConfiguration
	medialive: UpdateChannel
	medialive: UpdateChannelClass
	medialive: UpdateChannelPlacementGroup

Awalan layanan	Tindakan
	medialive: UpdateCloudWatchAlarmTemplate
	medialive: UpdateCloudWatchAlarmTemplateGroup
	medialive: UpdateCluster
	medialive: UpdateEventBridgeRuleTemplate
	medialive: UpdateEventBridgeRuleTemplateGroup
	medialive: UpdateInput
	medialive: UpdateInputDevice
	medialive: UpdateInputSecurityGroup
	medialive: UpdateMultiplex
	medialive: UpdateMultiplexProgram
	medialive: UpdateNetwork
	medialive: UpdateNode
	medialive: UpdateNodeState
	medialive: UpdateReservation
	medialive: UpdateSdiSource

Awalan layanan	Tindakan
mediatailor	mediatailor: ConfigureLogsForPlaybackConfiguration mediatailor: CreateChannel mediatailor: CreateLiveSource mediatailor: CreatePrefetchSchedule mediatailor: CreateProgram mediatailor: CreateSourceLocation mediatailor: CreateVodSource mediatailor: DeleteChannel mediatailor: DeleteChannelPolicy mediatailor: DeleteLiveSource mediatailor: DeletePlaybackConfiguration mediatailor: DeletePrefetchSchedule mediatailor: DeleteProgram mediatailor: DeleteSourceLocation mediatailor: DeleteVodSource mediatailor: DescribeChannel mediatailor: DescribeLiveSource mediatailor: DescribeProgram mediatailor: DescribeSourceLocation mediatailor: DescribeVodSource mediatailor: GetChannelPolicy

Awalan layanan	Tindakan
	mediatailor: GetChannelSchedule mediatailor: GetPlaybackConfiguration mediatailor: GetPrefetchSchedule mediatailor: ListAlerts mediatailor: ListChannels mediatailor: ListLiveSources mediatailor: ListPlaybackConfigurations mediatailor: ListPrefetchSchedules mediatailor: ListSourceLocations mediatailor: ListVodSources mediatailor: PutChannelPolicy mediatailor: PutPlaybackConfiguration mediatailor: StartChannel mediatailor: StopChannel mediatailor: UpdateChannel mediatailor: UpdateLiveSource mediatailor: UpdateProgram mediatailor: UpdateSourceLocation mediatailor: UpdateVodSource

Awalan layanan	Tindakan
memorydb	memorydb: BatchUpdateCluster
	memorydb: CopySnapshot
	memorydb: CreateAcl
	memorydb: CreateCluster
	memorydb: CreateMultiRegionCluster
	memorydb: CreateParameterGroup
	memorydb: CreateSnapshot
	memorydb: CreateSubnetGroup
	memorydb: CreateUser
	memorydb: DeleteAcl
	memorydb: DeleteCluster
	memorydb: DeleteMultiRegionCluster
	memorydb: DeleteParameterGroup
	memorydb: DeleteSnapshot
	memorydb: DeleteSubnetGroup
	memorydb: DeleteUser
	memorydb: DescribeAcls
	memorydb: DescribeClusters
	memorydb: DescribeEngineVersions
	memorydb: DescribeEvents
	memorydb: DescribeMultiRegionClusters

Awalan layanan	Tindakan
	memorydb: DescribeMultiRegionParameterGroups
	memorydb: DescribeMultiRegionParameters
	memorydb: DescribeParameterGroups
	memorydb: DescribeParameters
	memorydb: DescribeReservedNodes
	memorydb: DescribeReservedNodesOfferings
	memorydb: DescribeServiceUpdates
	memorydb: DescribeSnapshots
	memorydb: DescribeSubnetGroups
	memorydb: DescribeUsers
	memorydb: FailoverShard
	memorydb: ListAllowedMultiRegionClusterUpdates
	memorydb: ListAllowedNodeTypeUpdates
	memorydb: PurchaseReservedNodesOffering
	memorydb: ResetParameterGroup
	memorydb: UpdateAcl
	memorydb: UpdateCluster
	memorydb: UpdateMultiRegionCluster
	memorydb: UpdateParameterGroup
	memorydb: UpdateSubnetGroup
	memorydb: UpdateUser

Awalan layanan	Tindakan
mgh	mgh: AssociateCreatedArtifact mgh: AssociateDiscoveredResource mgh: AssociateSourceResource mgh: CreateHomeRegionControl mgh: CreateProgressUpdateStream mgh: DeleteHomeRegionControl mgh: DeleteProgressUpdateStream mgh: DescribeApplicationState mgh: DescribeHomeRegionControls mgh: DescribeMigrationTask mgh: DisassociateCreatedArtifact mgh: DisassociateDiscoveredResource mgh: DisassociateSourceResource mgh: GetHomeRegion mgh: ImportMigrationTask mgh: ListApplicationStates mgh: ListCreatedArtifacts mgh: ListDiscoveredResources mgh: ListMigrationTaskUpdates mgh: ListMigrationTasks mgh: ListProgressUpdateStreams

Awalan layanan	Tindakan
	mgh: ListSourceResources mgh: NotifyApplicationState mgh: NotifyMigrationTaskState mgh: PutResourceAttributes

Awalan layanan	Tindakan
mgn	mgn: ArchiveApplication
	mgn: ArchiveWave
	mgn: AssociateApplications
	mgn: AssociateSourceServers
	mgn: ChangeServerLifeCycleState
	mgn: CreateApplication
	mgn: CreateConnector
	mgn: CreateLaunchConfigurationTemplate
	mgn: CreateNetworkMigrationDefinition
	mgn: CreateReplicationConfigurationTemplate
	mgn: CreateWave
	mgn: DeleteApplication
	mgn: DeleteConnector
	mgn: DeleteJob
	mgn: DeleteLaunchConfigurationTemplate
	mgn: DeleteNetworkMigrationDefinition
	mgn: DeleteReplicationConfigurationTemplate
	mgn: DeleteSourceServer
	mgn: DeleteVcenterClient
	mgn: DeleteWave
	mgn: DescribeJobLogItems

Awalan layanan	Tindakan
	mgn: DescribeJobs
	mgn: DescribeLaunchConfigurationTemplates
	mgn: DescribeReplicationConfigurationTemplates
	mgn: DescribeVcenterClients
	mgn: DisassociateApplications
	mgn: DisassociateSourceServers
	mgn: DisconnectFromService
	mgn: FinalizeCutover
	mgn: GetNetworkMigrationDefinition
	mgn: GetNetworkMigrationMapperSegmentConstruct
	mgn: GetReplicationConfiguration
	mgn: InitializeService
	mgn: ListConnectors
	mgn: ListExportErrors
	mgn: ListExports
	mgn: ListImportErrors
	mgn: ListImportFileEnrichments
	mgn: ListImports
	mgn: ListManagedAccounts
	mgn: ListNetworkMigrationAnalyses
	mgn: ListNetworkMigrationAnalysisResults

Awalan layanan	Tindakan
	mgn: ListNetworkMigrationCodeGenerationSegments
	mgn: ListNetworkMigrationCodeGenerations
	mgn: ListNetworkMigrationDefinitions
	mgn: ListNetworkMigrationDeployedStacks
	mgn: ListNetworkMigrationDeployments
	mgn: ListNetworkMigrationExecutions
	mgn: ListNetworkMigrationMapperSegmentConstructs
	mgn: ListNetworkMigrationMapperSegments
	mgn: ListNetworkMigrationMappingUpdates
	mgn: ListNetworkMigrationMappings
	mgn: ListSourceServerActions
	mgn: ListTemplateActions
	mgn: MarkAsArchived
	mgn: PauseReplication
	mgn: PutSourceServerAction
	mgn: PutTemplateAction
	mgn: RemoveSourceServerAction
	mgn: RemoveTemplateAction
	mgn: ResumeReplication
	mgn: RetryDataReplication
	mgn: StartCutover

Awalan layanan	Tindakan
	mgn: StartExport
	mgn: StartImport
	mgn: StartImportFileEnrichment
	mgn: StartNetworkMigrationCodeGeneration
	mgn: StartNetworkMigrationMapping
	mgn: StartNetworkMigrationMappingUpdate
	mgn: StartReplication
	mgn: StartTest
	mgn: StopReplication
	mgn: TerminateTargetInstances
	mgn: UnarchiveApplication
	mgn: UnarchiveWave
	mgn: UpdateApplication
	mgn: UpdateConnector
	mgn: UpdateLaunchConfigurationTemplate
	mgn: UpdateNetworkMigrationDefinition
	mgn: UpdateNetworkMigrationMapperSegment
	mgn: UpdateReplicationConfiguration
	mgn: UpdateReplicationConfigurationTemplate
	mgn: UpdateSourceServer
	mgn: UpdateSourceServerReplicationType

Awalan layanan	Tindakan
	mgn: UpdateWave

Awalan layanan	Tindakan
Migrationhub-strategi	strategi hub migrasi: GetAntiPattern strategi hub migrasi: GetApplicationComponentDetails strategi hub migrasi: GetApplicationComponentStrategies strategi hub migrasi: GetAssessment strategi hub migrasi: GetImportFileTask strategi hub migrasi: GetLatestAssessmentId strategi hub migrasi: GetMessage strategi hub migrasi: GetPortfolioPreferences strategi hub migrasi: GetPortfolioSummary strategi hub migrasi: GetRecommendationReportDetails strategi hub migrasi: GetServerDetails strategi hub migrasi: GetServerStrategies strategi hub migrasi: ListAnalyzableServers strategi hub migrasi: ListAntiPatterns strategi hub migrasi: ListApplicationComponents strategi hub migrasi: ListCollectors strategi hub migrasi: ListImportFileTask strategi hub migrasi: ListJarArtifacts strategi hub migrasi: ListServers strategi hub migrasi: PutLogData strategi hub migrasi: PutMetricData

Awalan layanan	Tindakan
	strategi hub migrasi: PutPortfolioPreferences strategi hub migrasi: RegisterCollector strategi hub migrasi: SendMessage strategi hub migrasi: StartAssessment strategi hub migrasi: StartImportFileTask strategi hub migrasi: StartRecommendationReportGeneration strategi hub migrasi: StopAssessment strategi hub migrasi: UpdateApplicationComponentConfig strategi hub migrasi: UpdateCollectorConfiguration strategi hub migrasi: UpdateServerConfig

Awalan layanan	Tindakan
penargetan seluler	penargetan seluler: CreateApp
	penargetan seluler: CreateCampaign
	penargetan seluler: CreateEmailTemplate
	penargetan seluler: CreateExportJob
	penargetan seluler: CreateImportJob
	penargetan seluler: CreateInAppTemplate
	penargetan seluler: CreateJourney
	penargetan seluler: CreatePushTemplate
	penargetan seluler: CreateRecommenderConfiguration
	penargetan seluler: CreateSegment
	penargetan seluler: CreateSmsTemplate
	penargetan seluler: CreateVoiceTemplate
	penargetan seluler: DeleteAdmChannel
	penargetan seluler: DeleteApnsChannel
	penargetan seluler: DeleteApnsSandboxChannel
	penargetan seluler: DeleteApnsVoipChannel
	penargetan seluler: DeleteApnsVoipSandboxChannel
	penargetan seluler: DeleteApp
	penargetan seluler: DeleteBaiduChannel
	penargetan seluler: DeleteCampaign
	penargetan seluler: DeleteEmailChannel

Awalan layanan	Tindakan
	penargetan seluler: DeleteEmailTemplate
	penargetan seluler: DeleteEndpoint
	penargetan seluler: DeleteEventStream
	penargetan seluler: DeleteGcmChannel
	penargetan seluler: DeleteInAppTemplate
	penargetan seluler: DeleteJourney
	penargetan seluler: DeletePushTemplate
	penargetan seluler: DeleteRecommenderConfiguration
	penargetan seluler: DeleteSegment
	penargetan seluler: DeleteSmsChannel
	penargetan seluler: DeleteSmsTemplate
	penargetan seluler: DeleteUserEndpoints
	penargetan seluler: DeleteVoiceChannel
	penargetan seluler: DeleteVoiceTemplate
	penargetan seluler: GetAdmChannel
	penargetan seluler: GetApnsChannel
	penargetan seluler: GetApnsSandboxChannel
	penargetan seluler: GetApnsVoipChannel
	penargetan seluler: GetApnsVoipSandboxChannel
	penargetan seluler: GetApp
	penargetan seluler: GetApplicationDateRangeKpi

Awalan layanan	Tindakan
	penargetan seluler: GetApplicationSettings
	penargetan seluler: GetApps
	penargetan seluler: GetBaiduChannel
	penargetan seluler: GetCampaign
	penargetan seluler: GetCampaignActivities
	penargetan seluler: GetCampaignDateRangeKpi
	penargetan seluler: GetCampaignVersion
	penargetan seluler: GetCampaignVersions
	penargetan seluler: GetCampaigns
	penargetan seluler: GetChannels
	penargetan seluler: GetEmailChannel
	penargetan seluler: GetEmailTemplate
	penargetan seluler: GetEndpoint
	penargetan seluler: GetEventStream
	penargetan seluler: GetExportJob
	penargetan seluler: GetExportJobs
	penargetan seluler: GetGcmChannel
	penargetan seluler: GetImportJob
	penargetan seluler: GetImportJobs
	penargetan seluler: GetInAppMessages
	penargetan seluler: GetInAppTemplate

Awalan layanan	Tindakan
	penargetan seluler: GetJourney
	penargetan seluler: GetJourneyDateRangeKpi
	penargetan seluler: GetJourneyExecutionActivityMetrics
	penargetan seluler: GetJourneyExecutionMetrics
	penargetan seluler: GetJourneyRunExecutionActivityMetrics
	penargetan seluler: GetJourneyRunExecutionMetrics
	penargetan seluler: GetJourneyRuns
	penargetan seluler: GetPushTemplate
	penargetan seluler: GetRecommenderConfiguration
	penargetan seluler: GetRecommenderConfigurations
	penargetan seluler: GetSegment
	penargetan seluler: GetSegmentExportJobs
	penargetan seluler: GetSegmentImportJobs
	penargetan seluler: GetSegmentVersion
	penargetan seluler: GetSegmentVersions
	penargetan seluler: GetSegments
	penargetan seluler: GetSmsChannel
	penargetan seluler: GetSmsTemplate
	penargetan seluler: GetUserEndpoints
	penargetan seluler: GetVoiceChannel
	penargetan seluler: GetVoiceTemplate

Awalan layanan	Tindakan
	penargetan seluler: ListJourneys
	penargetan seluler: ListTemplateVersions
	penargetan seluler: ListTemplates
	penargetan seluler: PhoneNumberValidate
	penargetan seluler: PutEventStream
	penargetan seluler: RemoveAttributes
	penargetan seluler: UpdateAdmChannel
	penargetan seluler: UpdateApnsChannel
	penargetan seluler: UpdateApnsSandboxChannel
	penargetan seluler: UpdateApnsVoipChannel
	penargetan seluler: UpdateApnsVoipSandboxChannel
	penargetan seluler: UpdateApplicationSettings
	penargetan seluler: UpdateBaiduChannel
	penargetan seluler: UpdateCampaign
	penargetan seluler: UpdateEmailChannel
	penargetan seluler: UpdateEmailTemplate
	penargetan seluler: UpdateEndpoint
	penargetan seluler: UpdateEndpointsBatch
	penargetan seluler: UpdateGcmChannel
	penargetan seluler: UpdateInAppTemplate
	penargetan seluler: UpdateJourney

Awalan layanan	Tindakan
	penargetan seluler: UpdateJourneyState
	penargetan seluler: UpdatePushTemplate
	penargetan seluler: UpdateRecommenderConfiguration
	penargetan seluler: UpdateSegment
	penargetan seluler: UpdateSmsChannel
	penargetan seluler: UpdateSmsTemplate
	penargetan seluler: UpdateTemplateActiveVersion
	penargetan seluler: UpdateVoiceChannel
	penargetan seluler: UpdateVoiceTemplate
	Penargetan Mobile:VerifyotpMessage

Awalan layanan	Tindakan
mq	mq: CreateBroker mq: CreateConfiguration mq: CreateUser mq: DeleteBroker mq: DeleteConfiguration mq: DeleteUser mq: DescribeBroker mq: DescribeBrokerEngineTypes mq: DescribeBrokerInstanceOptions mq: DescribeConfiguration mq: DescribeConfigurationRevision mq: DescribeSharedResources mq: DescribeUser mq: ListBrokers mq: ListConfigurationRevisions mq: ListConfigurations mq: ListUsers MQ: Mempromosikan mq: RebootBroker mq: UpdateBroker mq: UpdateConfiguration

Awalan layanan	Tindakan
	mq: UpdateUser

Awalan layanan	Tindakan
manajer jaringan	manajer jaringan: AcceptAttachment
	manajer jaringan: AssociateConnectPeer
	manajer jaringan: AssociateCustomerGateway
	manajer jaringan: AssociateLink
	manajer jaringan: AssociateTransitGatewayConnectPeer
	manajer jaringan: CreateConnectAttachment
	manajer jaringan: CreateConnectPeer
	manajer jaringan: CreateConnection
	manajer jaringan: CreateCoreNetwork
	manajer jaringan: CreateDevice
	manajer jaringan: CreateDirectConnectGatewayAttachment
	manajer jaringan: CreateGlobalNetwork
	manajer jaringan: CreateLink
	manajer jaringan: CreateSite
	manajer jaringan: CreateSiteToSiteVpnAttachment
	manajer jaringan: CreateTransitGatewayPeering
	manajer jaringan: CreateTransitGatewayRouteTableAttachment
	manajer jaringan: CreateVpcAttachment
	manajer jaringan: DeleteAttachment
	manajer jaringan: DeleteConnectPeer
	manajer jaringan: DeleteConnection

Awalan layanan	Tindakan
	manajer jaringan: DeleteCoreNetwork
	manajer jaringan: DeleteCoreNetworkPolicyVersion
	manajer jaringan: DeleteDevice
	manajer jaringan: DeleteGlobalNetwork
	manajer jaringan: DeleteLink
	manajer jaringan: DeletePeering
	manajer jaringan: DeleteResourcePolicy
	manajer jaringan: DeleteSite
	manajer jaringan: DeregisterTransitGateway
	manajer jaringan: DescribeGlobalNetworks
	manajer jaringan: DisassociateConnectPeer
	manajer jaringan: DisassociateCustomerGateway
	manajer jaringan: DisassociateLink
	manajer jaringan: DisassociateTransitGatewayConnectPeer
	manajer jaringan: ExecuteCoreNetworkChangeSet
	manajer jaringan: GetConnectAttachment
	manajer jaringan: GetConnectPeer
	manajer jaringan: GetConnectPeerAssociations
	manajer jaringan: GetConnections
	manajer jaringan: GetCoreNetwork
	manajer jaringan: GetCoreNetworkChangeEvents

Awalan layanan	Tindakan
	manajer jaringan: GetCoreNetworkChangeSet
	manajer jaringan: GetCoreNetworkPolicy
	manajer jaringan: GetCustomerGatewayAssociations
	manajer jaringan: GetDevices
	manajer jaringan: GetLinkAssociations
	manajer jaringan: GetLinks
	manajer jaringan: GetNetworkResourceCounts
	manajer jaringan: GetNetworkResourceRelationships
	manajer jaringan: GetNetworkResources
	manajer jaringan: GetNetworkRoutes
	manajer jaringan: GetNetworkTelemetry
	manajer jaringan: GetResourcePolicy
	manajer jaringan: GetRouteAnalysis
	manajer jaringan: GetSiteToSiteVpnAttachment
	manajer jaringan: GetSites
	manajer jaringan: GetTransitGatewayConnectPeerAssociations
	manajer jaringan: GetTransitGatewayPeering
	manajer jaringan: GetTransitGatewayRegistrations
	manajer jaringan: GetTransitGatewayRouteTableAttachment
	manajer jaringan: GetVpcAttachment
	manajer jaringan: ListAttachmentRoutingPolicyAssociations

Awalan layanan	Tindakan
	manajer jaringan: ListAttachments
	manajer jaringan: ListConnectPeers
	manajer jaringan: ListCoreNetworkPolicyVersions
	manajer jaringan: ListCoreNetworkPrefixListAssociations
	manajer jaringan: ListCoreNetworkRoutingInformation
	manajer jaringan: ListCoreNetworks
	manajer jaringan: ListOrganizationServiceAccessStatus
	manajer jaringan: ListPeerings
	manajer jaringan: PutAttachmentRoutingPolicyLabel
	manajer jaringan: PutCoreNetworkPolicy
	manajer jaringan: PutResourcePolicy
	manajer jaringan: RegisterTransitGateway
	manajer jaringan: RejectAttachment
	manajer jaringan: RemoveAttachmentRoutingPolicyLabel
	manajer jaringan: RestoreCoreNetworkPolicyVersion
	manajer jaringan: StartOrganizationServiceAccessUpdate
	manajer jaringan: StartRouteAnalysis
	manajer jaringan: UpdateConnection
	manajer jaringan: UpdateCoreNetwork
	manajer jaringan: UpdateDevice
	manajer jaringan: UpdateDirectConnectGatewayAttachment

Awalan layanan	Tindakan
	manajer jaringan: UpdateGlobalNetwork manajer jaringan: UpdateLink manajer jaringan: UpdateNetworkResourceMetadata manajer jaringan: UpdateSite manajer jaringan: UpdateVpcAttachment

Awalan layanan	Tindakan
gesit	gesit: AcceptEulas gesit: CreateLaunchProfile gesit: CreateStreamingImage gesit: CreateStreamingSession gesit: CreateStreamingSessionStream gesit: CreateStudio gesit: CreateStudioComponent gesit: DeleteLaunchProfile gesit: DeleteLaunchProfileMember gesit: DeleteStreamingImage gesit: DeleteStreamingSession gesit: DeleteStudio gesit: DeleteStudioComponent gesit: DeleteStudioMember gesit: GetEula gesit: GetLaunchProfileDetails gesit: GetStreamingImage gesit: GetStreamingSession gesit: GetStreamingSessionBackup gesit: GetStreamingSessionStream gesit: GetStudio

Awalan layanan	Tindakan
	gesit: GetStudioComponent gesit: GetStudioMember gesit: ListEulas gesit: ListLaunchProfileMembers gesit: ListLaunchProfiles gesit: ListStreamingImages gesit: ListStreamingSessionBackups gesit: ListStreamingSessions gesit: ListStudioComponents gesit: ListStudioMembers gesit: ListStudios gesit: PutLaunchProfileMembers gesit: PutStudioMembers gesit: StartStreamingSession gesit: StartStudioSSOConfigurationRepair gesit: StopStreamingSession gesit: UpdateLaunchProfile gesit: UpdateLaunchProfileMember gesit: UpdateStreamingImage gesit: UpdateStudio gesit: UpdateStudioComponent

Awalan layanan	Tindakan
omics	omics: AbortMultipartReadSetUpload
	omics: AcceptShare
	omics: BatchDeleteReadSet
	omics: CancelAnnotationImportJob
	omics: CancelRun
	omics: CancelRunBatch
	omics: CancelVariantImportJob
	omics: CompleteMultipartReadSetUpload
	omics: CreateAnnotationStore
	omics: CreateAnnotationStoreVersion
	omics: CreateConfiguration
	omics: CreateMultipartReadSetUpload
	omics: CreateReferenceStore
	omics: CreateRunGroup
	omics: CreateSequenceStore
	omics: CreateShare
	omics: CreateVariantStore
	omics: CreateWorkflow
	omics: CreateWorkflowVersion
	omics: DeleteAnnotationStore
	omics: DeleteAnnotationStoreVersions

Awalan layanan	Tindakan
	omics: DeleteBatch
	omics: DeleteConfiguration
	omics: DeleteReference
	omics: DeleteReferenceStore
	omics: DeleteRun
	omics: DeleteRunBatch
	omics: DeleteRunGroup
	omics: DeleteSequenceStore
	omics: DeleteShare
	omics: DeleteVariantStore
	omics: DeleteWorkflow
	omics: DeleteWorkflowVersion
	omics: GetAnnotationImportJob
	omics: GetAnnotationStore
	omics: GetAnnotationStoreVersion
	omics: GetBatch
	omics: GetConfiguration
	omics: GetReadSet
	omics: GetReadSetActivationJob
	omics: GetReadSetExportJob
	omics: GetReadSetImportJob

Awalan layanan	Tindakan
	omics: GetReadSetMetadata
	omics: GetReference
	omics: GetReferenceImportJob
	omics: GetReferenceMetadata
	omics: GetReferenceStore
	omics: GetRun
	omics: GetRunGroup
	omics: GetRunTask
	omics: GetSequenceStore
	omics: GetShare
	omics: GetVariantImportJob
	omics: GetVariantStore
	omics: GetWorkflow
	omics: GetWorkflowVersion
	omics: ListAnnotationImportJobs
	omics: ListAnnotationStoreVersions
	omics: ListAnnotationStores
	omics: ListBatch
	omics: ListConfigurations
	omics: ListMultipartReadSetUploads
	omics: ListReadSetActivationJobs

Awalan layanan	Tindakan
	omics: ListReadSetExportJobs
	omics: ListReadSetImportJobs
	omics: ListReadSetUploadParts
	omics: ListReadSets
	omics: ListReferenceImportJobs
	omics: ListReferenceStores
	omics: ListReferences
	omics: ListRunGroups
	omics: ListRunTasks
	omics: ListRuns
	omics: ListRunsInBatch
	omics: ListSequenceStores
	omics: ListShares
	omics: ListVariantImportJobs
	omics: ListVariantStores
	omics: ListWorkflowVersions
	omics: ListWorkflows
	omics: StartAnnotationImportJob
	omics: StartReadSetActivationJob
	omics: StartReadSetExportJob
	omics: StartReadSetImportJob

Awalan layanan	Tindakan
	omics: StartReferenceImportJob
	omics: StartRun
	omics: StartRunBatch
	omics: StartVariantImportJob
	omics: UpdateAnnotationStore
	omics: UpdateAnnotationStoreVersion
	omics: UpdateRunGroup
	omics: UpdateVariantStore
	omics: UpdateWorkflow
	omics: UpdateWorkflowVersion
	omics: UploadReadSetPart

Awalan layanan	Tindakan
opsworks	opsworks: AssignInstance opsworks: AssignVolume opsworks: AssociateElasticIp opsworks: AttachElasticLoadBalancer opsworks: CloneStack opsworks: CreateApp opsworks: CreateDeployment opsworks: CreateInstance opsworks: CreateLayer opsworks: CreateStack opsworks: CreateUserProfile opsworks: DeleteApp opsworks: DeleteInstance opsworks: DeleteLayer opsworks: DeleteStack opsworks: DeleteUserProfile opsworks: DeregisterEcsCluster opsworks: DeregisterElasticIp opsworks: DeregisterInstance opsworks: DeregisterRdsDbInstance opsworks: DeregisterVolume

Awalan layanan	Tindakan
	opsworks: DescribeAgentVersions opsworks: DescribeApps opsworks: DescribeCommands opsworks: DescribeDeployments opsworks: DescribeEcsClusters opsworks: DescribeElasticIps opsworks: DescribeElasticLoadBalancers opsworks: DescribeInstances opsworks: DescribeLayers opsworks: DescribeLoadBasedAutoScaling opsworks: DescribeMyUserProfile opsworks: DescribeOperatingSystems opsworks: DescribePermissions opsworks: DescribeRaidArrays opsworks: DescribeRdsDbInstances opsworks: DescribeServiceErrors opsworks: DescribeStackProvisioningParameters opsworks: DescribeStackSummary opsworks: DescribeStacks opsworks: DescribeTimeBasedAutoScaling opsworks: DescribeUserProfiles

Awalan layanan	Tindakan
	opsworks: DescribeVolumes opsworks: DetachElasticLoadBalancer opsworks: DisassociateElasticIp opsworks: GetHostnameSuggestion opsworks: GrantAccess opsworks: RebootInstance opsworks: RegisterEcsCluster opsworks: RegisterElasticIp opsworks: RegisterInstance opsworks: RegisterRdsDbInstance opsworks: RegisterVolume opsworks: SetLoadBasedAutoScaling opsworks: SetPermission opsworks: SetTimeBasedAutoScaling opsworks: StartInstance opsworks: StartStack opsworks: StopInstance opsworks: StopStack opsworks: UnassignInstance opsworks: UnassignVolume opsworks: UpdateApp

Awalan layanan	Tindakan
	opsworks: UpdateElasticIp opsworks: UpdateInstance opsworks: UpdateLayer opsworks: UpdateMyUserProfile opsworks: UpdateRdsDbInstance opsworks: UpdateStack opsworks: UpdateUserProfile opsworks: UpdateVolume

Awalan layanan	Tindakan
opsworks-cm	opsworks-cm: AssociateNode
	opsworks-cm: CreateBackup
	opsworks-cm: CreateServer
	opsworks-cm: DeleteBackup
	opsworks-cm: DeleteServer
	opsworks-cm: DescribeAccountAttributes
	opsworks-cm: DescribeBackups
	opsworks-cm: DescribeEvents
	opsworks-cm: DescribeNodeAssociationStatus
	opsworks-cm: DescribeServers
	opsworks-cm: DisassociateNode
	opsworks-cm: ExportServerEngineAttribute
	opsworks-cm: RestoreServer
	opsworks-cm: StartMaintenance
	opsworks-cm: UpdateServer
	opsworks-cm: UpdateServerEngineAttributes

Awalan layanan	Tindakan
organisasi	organisasi: AcceptHandshake organisasi: AttachPolicy organisasi: CancelHandshake organisasi: CloseAccount organisasi: CreateAccount organisasi: CreateGovCloudAccount organisasi: CreateOrganization organisasi: CreateOrganizationalUnit organisasi: CreatePolicy organisasi: DeclineHandshake organisasi: DeleteOrganization organisasi: DeleteOrganizationalUnit organisasi: DeletePolicy organisasi: DeleteResourcePolicy organisasi: DeregisterDelegatedAdministrator organisasi: DescribeAccount organisasi: DescribeCreateAccountStatus organisasi: DescribeEffectivePolicy organisasi: DescribeHandshake organisasi: DescribeOrganization organisasi: DescribeOrganizationalUnit

Awalan layanan	Tindakan
	organisasi: DescribePolicy organisasi: DescribeResourcePolicy organisasi: DescribeResponsibilityTransfer organisasi: DetachPolicy organisasi: DisableAWSServiceAccess organisasi: DisablePolicyType organisasi: EnableAWSServiceAccess organisasi: EnableAllFeatures organisasi: EnablePolicyType organisasi: InviteAccountToOrganization organisasi: LeaveOrganization organisasi: ListAWSServiceAccessForOrganization organisasi: ListAccounts organisasi: ListAccountsForParent organisasi: ListAccountsWithInvalidEffectivePolicy organisasi: ListChildren organisasi: ListCreateAccountStatus organisasi: ListDelegatedAdministrators organisasi: ListDelegatedServicesForAccount organisasi: ListHandshakesForAccount organisasi: ListHandshakesForOrganization

Awalan layanan	Tindakan
	organisasi: ListInboundResponsibilityTransfers organisasi: ListOrganizationalUnitsForParent organisasi: ListOutboundResponsibilityTransfers organisasi: ListParents organisasi: ListPolicies organisasi: ListPoliciesForTarget organisasi: ListRoots organisasi: ListTargetsForPolicy organisasi: MoveAccount organisasi: PutResourcePolicy organisasi: RegisterDelegatedAdministrator organisasi: RemoveAccountFromOrganization organisasi: TerminateResponsibilityTransfer organisasi: UpdateOrganizationalUnit organisasi: UpdatePolicy organisasi: UpdateResponsibilityTransfer

Awalan layanan	Tindakan
pos terdepan	pos terdepan: CancelCapacityTask
	pos terdepan: CancelOrder
	pos terdepan: CreateOrder
	pos terdepan: CreateOutpost
	pos terdepan: CreatePrivateConnectivityConfig
	pos terdepan: CreateQuote
	pos terdepan: CreateRenewal
	pos terdepan: CreateSite
	pos terdepan: DeleteOutpost
	pos terdepan: DeleteQuote
	pos terdepan: DeleteSite
	pos terdepan: GetCapacityTask
	pos terdepan: GetCatalogItem
	pos terdepan: GetConnection
	pos terdepan: GetOrder
	pos terdepan: GetOutpost
	pos terdepan: GetOutpostBillingInformation
	pos terdepan: GetOutpostInstanceTypes
	pos terdepan: GetOutpostSupportedInstanceTypes
	pos terdepan: GetPrivateConnectivityConfig
	pos terdepan: GetQuote

Awalan layanan	Tindakan
	pos terdepan: GetRenewalPricing
	pos terdepan: GetSite
	pos terdepan: GetSiteAddress
	pos terdepan: ListAssetInstances
	pos terdepan: ListAssets
	pos terdepan: ListBlockingInstancesForCapacityTask
	pos terdepan: ListCapacityTasks
	pos terdepan: ListCatalogItems
	pos terdepan: ListOrderableInstanceTypes
	pos terdepan: ListOrders
	pos terdepan: ListOutposts
	pos terdepan: ListQuotes
	pos terdepan: ListSites
	pos terdepan: StartCapacityTask
	pos terdepan: StartConnection
	pos terdepan: UpdateOutpost
	pos terdepan: UpdateQuote
	pos terdepan: UpdateSite
	pos terdepan: UpdateSiteAddress
	pos terdepan: UpdateSiteRackPhysicalProperties

Awalan layanan	Tindakan
pi	pi: CreatePerformanceAnalysisReport pi: DeletePerformanceAnalysisReport pi: DescribeDimensionKeys pi: GetDimensionKeyDetails pi: GetPerformanceAnalysisReport pi: GetResourceMetadata pi: GetResourceMetrics pi: ListAvailableResourceDimensions pi: ListAvailableResourceMetrics pi: ListPerformanceAnalysisReports
pipa	pipa: CreatePipe pipa: DeletePipe pipa: DescribePipe pipa: ListPipes pipa: StartPipe pipa: StopPipe pipa: UpdatePipe

Awalan layanan	Tindakan
polly	polly: DeleteLexicon polly: DescribeVoices polly: GetLexicon polly: GetSpeechSynthesisTask polly: ListLexicons polly: ListSpeechSynthesisTasks polly: PutLexicon polly: StartSpeechSynthesisTask polly: SynthesizeSpeech

Awalan layanan	Tindakan
profile	Profil: AddProfileKey
	Profil: BatchGetCalculatedAttributeForProfile
	Profil: BatchGetProfile
	Profil: CreateCalculatedAttributeDefinition
	Profil: CreateDomain
	Profil: CreateEventStream
	Profil: CreateProfile
	Profil: CreateRecommender
	Profil: CreateRecommenderFilter
	Profil: CreateRecommenderSchema
	Profil: CreateSegmentDefinition
	Profil: CreateSegmentEstimate
	Profil: CreateSegmentSnapshot
	Profil: CreateUploadJob
	Profil: DeleteCalculatedAttributeDefinition
	Profil: DeleteDomain
	Profil: DeleteDomainObjectType
	Profil: DeleteEventStream
	Profil: DeleteIntegration
	Profil: DeleteProfile
	Profil: DeleteProfileKey

Awalan layanan	Tindakan
	Profil: DeleteProfileObject
	Profil: DeleteProfileObjectType
	Profil: DeleteRecommender
	Profil: DeleteRecommenderFilter
	Profil: DeleteRecommenderSchema
	Profil: DeleteSegmentDefinition
	Profil: DeleteWorkflow
	Profil: DetectProfileObjectType
	Profil: GetAutoMergingPreview
	Profil: GetCalculatedAttributeDefinition
	Profil: GetCalculatedAttributeForProfile
	Profil: GetDomain
	Profil: GetDomainObjectType
	Profil: GetEventStream
	Profil: GetIdentityResolutionJob
	Profil: GetIntegration
	Profil: GetMatches
	Profil: GetObjectTypeAttributeStatistics
	Profil: GetProfileObjectType
	Profil: GetProfileObjectTypeTemplate
	Profil: GetProfileRecommendations

Awalan layanan	Tindakan
	Profil: GetRecommender
	Profil: GetRecommenderFilter
	Profil: GetRecommenderSchema
	Profil: GetSegmentDefinition
	Profil: GetSegmentEstimate
	Profil: GetSegmentMembership
	Profil: GetSegmentSnapshot
	Profil: GetSimilarProfiles
	Profil: GetUploadJob
	Profil: GetUploadJobPath
	Profil: GetWorkflow
	Profil: GetWorkflowSteps
	Profil: ListAccountIntegrations
	Profil: ListCalculatedAttributeDefinitions
	Profil: ListCalculatedAttributesForProfile
	Profil: ListDomainLayouts
	Profil: ListDomainObjectTypes
	Profil: ListDomains
	Profil: ListEventStreams
	Profil: ListIdentityResolutionJobs
	Profil: ListIntegrations

Awalan layanan	Tindakan
	Profil: ListObjectTypeAttributeValues
	Profil: ListObjectTypeAttributes
	Profil: ListProfileAttributeValues
	Profil: ListProfileObjectTypeTemplates
	Profil: ListProfileObjectTypes
	Profil: ListProfileObjects
	Profil: ListRecommenderFilters
	Profil: ListRecommenderRecipes
	Profil: ListRecommenderSchemas
	Profil: ListRecommenders
	Profil: ListRuleBasedMatches
	Profil: ListSegmentDefinitions
	Profil: ListUploadJobs
	Profil: ListWorkflows
	Profil: MergeProfiles
	Profil: PutDomainObjectType
	Profil: PutIntegration
	Profil: PutProfileObject
	Profil: PutProfileObjectType
	Profil: SearchProfiles
	Profil: StartRecommender

Awalan layanan	Tindakan
	Profil: StartUploadJob Profil: StopRecommender Profil: StopUploadJob Profil: UpdateCalculatedAttributeDefinition Profil: UpdateDomain Profil: UpdateProfile Profil: UpdateRecommender

Awalan layanan	Tindakan
qldb	qldb: CancelJournalKinesisStream qldb: CreateLedger qldb: DeleteLedger qldb: DescribeJournalKinesisStream qldb: DescribeJournalS3Export qldb: DescribeLedger qldb: ExportJournalToS3 qldb: GetBlock qldb: GetDigest qldb: GetRevision qldb: ListJournalKinesisStreamsForLedger qldb: ListJournalS3Exports qldb: ListJournalS3ExportsForLedger qldb: ListLedgers qldb: StreamJournalToKinesis qldb: UpdateLedger qldb: UpdateLedgerPermissionsMode

Awalan layanan	Tindakan
ram	ram: AcceptResourceShareInvitation
	ram: AssociateResourceShare
	ram: AssociateResourceSharePermission
	ram: CreatePermission
	ram: CreatePermissionVersion
	ram: CreateResourceShare
	ram: DeletePermission
	ram: DeletePermissionVersion
	ram: DeleteResourceShare
	ram: DisassociateResourceShare
	ram: DisassociateResourceSharePermission
	ram: EnableSharingWithAwsOrganization
	ram: GetPermission
	ram: GetResourcePolicies
	ram: GetResourceShareAssociations
	ram: GetResourceShareInvitations
	ram: GetResourceShares
	ram: ListPendingInvitationResources
	ram: ListPermissionAssociations
	domba jantan: ListPermissionVersions
	domba jantan: ListPermissions

Awalan layanan	Tindakan
	domba jantan: ListPrincipals domba jantan: ListReplacePermissionAssociationsWork domba jantan: ListResourceSharePermissions domba jantan: ListResourceTypes domba jantan: ListResources domba jantan: ListSourceAssociations domba jantan: PromotePermissionCreatedFromPolicy domba jantan: PromoteResourceShareCreatedFromPolicy domba jantan: RejectResourceShareInvitation domba jantan: ReplacePermissionAssociations domba jantan: SetDefaultPermissionVersion domba jantan: UpdateResourceShare
rbin	rbin: CreateRule rbin: DeleteRule rbin: GetRule rbin: ListRules rbin: LockRule rbin: UnlockRule rbin: UpdateRule

Awalan layanan	Tindakan
rds	rds: AddRoleToDBCluster
	rds: AddRoleToDBInstance
	rds: AddSourceIdentifierToSubscription
	rds: ApplyPendingMaintenanceAction
	rds: AuthorizeDBSecurityGroupIngress
	RDS: BacktrackDBCluster
	rds: CancelExportTask
	rds: CopyDBClusterParameterGroup
	rds: CopyDBClusterSnapshot
	rds: CopyDBParameterGroup
	RDS: CopyDBSnapshot
	rds: CopyOptionGroup
	rds: CreateCustomDBEngineVersion
	rds: CreateDBClusterParameterGroup
	rds: CreateDBParameterGroup
	RDS: dibuatBProxy
	rds: CreateDBProxyEndpoint
	rds: CreateDBSecurityGroup
	rds: CreateDBSubnetGroup
	rds: CreateEventSubscription
	rds: CreateGlobalCluster

Awalan layanan	Tindakan
	rds: CreateOptionGroup
	rds: DeleteBlueGreenDeployment
	rds: DeleteDBClusterAutomatedBackup
	rds: DeleteDBClusterParameterGroup
	rds: DeleteDBClusterSnapshot
	rds: DeleteDBInstanceAutomatedBackup
	rds: DeleteDBParameterGroup
	RDS: DeletedBProxy
	rds: DeleteDBProxyEndpoint
	rds: DeleteDBSecurityGroup
	RDS: DeletedBSnapshot
	rds: DeleteDBSubnetGroup
	rds: DeleteEventSubscription
	rds: DeleteGlobalCluster
	rds: DeleteOptionGroup
	rds: DeregisterDBProxyTargets
	rds: DescribeAccountAttributes
	rds: DescribeBlueGreenDeployments
	rds: DescribeCertificates
	rds: DescribeDBClusterAutomatedBackups
	rds: DescribeDBClusterBacktracks

Awalan layanan	Tindakan
	rds: DescribeDBClusterEndpoints
	rds: DescribeDBClusterParameterGroups
	rds: DescribeDBClusterParameters
	rds: DescribeDBClusterSnapshotAttributes
	rds: DescribeDBClusterSnapshots
	RDS: DescribedBClusters
	rds: DescribeDBEngineVersions
	rds: DescribeDBInstanceAutomatedBackups
	RDS: DescribedBinstances
	rds: DescribeDBLogFiles
	rds: DescribeDBMajorEngineVersions
	rds: DescribeDBParameterGroups
	RDS: DescribedBParameters
	RDS: DescribedBProxies
	rds: DescribeDBProxyEndpoints
	rds: DescribeDBProxyTargetGroups
	rds: DescribeDBProxyTargets
	RDS:DescribedBrecommendations
	rds: DescribeDBSecurityGroups
	rds: DescribeDBSnapshotAttributes
	rds: DescribeDBSnapshotTenantDatabases

Awalan layanan	Tindakan
	RDS: DescribedBSnapshots
	rds: DescribeDBSubnetGroups
	rds: DescribeEngineDefaultClusterParameters
	rds: DescribeEngineDefaultParameters
	rds: DescribeEventCategories
	rds: DescribeEventSubscriptions
	rds: DescribeEvents
	rds: DescribeExportTasks
	rds: DescribeGlobalClusters
	rds: DescribeIntegrations
	rds: DescribeOptionGroupOptions
	rds: DescribeOptionGroups
	rds: DescribeOrderableDBInstanceOptions
	rds: DescribePendingMaintenanceActions
	rds: DescribeReservedDBInstances
	rds: DescribeReservedDBInstancesOfferings
	rds: DescribeSourceRegions
	rds: DescribeTenantDatabases
	rds: DescribeValidDBInstanceModifications
	rds: DownloadCompleteDBLogFile
	rds: DownloadDBLogFilePortion

Awalan layanan	Tindakan
	RDS: FailoverDBCluster
	rds: FailoverGlobalCluster
	rds: ModifyActivityStream
	rds: ModifyCertificates
	rds: ModifyCurrentDBClusterCapacity
	rds: ModifyDBClusterEndpoint
	rds: ModifyDBClusterParameterGroup
	rds: ModifyDBClusterSnapshotAttribute
	rds: ModifyDBParameterGroup
	RDS: ModifyDBProxy
	rds: ModifyDBProxyEndpoint
	rds: ModifyDBProxyTargetGroup
	RDS: ModifydbreCommendation
	RDS: ModifyDBSnapshot
	rds: ModifyDBSnapshotAttribute
	rds: ModifyDBSubnetGroup
	rds: ModifyEventSubscription
	rds: ModifyGlobalCluster
	rds: ModifyOptionGroup
	rds: ModifyTenantDatabase
	rds: PurchaseReservedDBInstancesOffering

Awalan layanan	Tindakan
	RDS: RebootDBCluster
	rds: RegisterDBProxyTargets
	rds: RemoveFromGlobalCluster
	rds: RemoveRoleFromDBCluster
	rds: RemoveRoleFromDBInstance
	rds: RemoveSourceIdentifierFromSubscription
	rds: ResetDBClusterParameterGroup
	rds: ResetDBParameterGroup
	rds: RestoreDBClusterFromS3
	rds: RestoreDBClusterFromSnapshot
	rds: RestoreDBClusterToPointInTime
	rds: RestoreDBInstanceFromDBSnapshot
	rds: RestoreDBInstanceFromS3
	rds: RestoreDBInstanceToPointInTime
	rds: RevokeDBSecurityGroupIngress
	rds: StartActivityStream
	RDS: StartDBCluster
	RDS: StartDBInstance
	rds: StartDBInstanceAutomatedBackupsReplication
	rds: StartExportTask
	rds: StopActivityStream

Awalan layanan	Tindakan
	RDS: STOPDBCluster RDS: StopdbInstance rds: StopDBInstanceAutomatedBackupsReplication rds: SwitchoverBlueGreenDeployment rds: SwitchoverGlobalCluster rds: SwitchoverReadReplica

Awalan layanan	Tindakan
redshift	pergeseran merah: AcceptReservedNodeExchange
	pergeseran merah: AddPartner
	pergeseran merah: AssociateDataShareConsumer
	pergeseran merah: AuthorizeClusterSecurityGroupIngress
	pergeseran merah: AuthorizeDataShare
	pergeseran merah: AuthorizeEndpointAccess
	pergeseran merah: AuthorizeSnapshotAccess
	pergeseran merah: BatchDeleteClusterSnapshots
	pergeseran merah: BatchModifyClusterSnapshots
	pergeseran merah: CancelResize
	pergeseran merah: CopyClusterSnapshot
	pergeseran merah: CreateAuthenticationProfile
	pergeseran merah: CreateCluster
	pergeseran merah: CreateClusterParameterGroup
	pergeseran merah: CreateClusterSecurityGroup
	pergeseran merah: CreateClusterSnapshot
	pergeseran merah: CreateClusterSubnetGroup
	pergeseran merah: CreateCustomDomainAssociation
	pergeseran merah: CreateEndpointAccess
	pergeseran merah: CreateEventSubscription
	pergeseran merah: CreateHsmClientCertificate

Awalan layanan	Tindakan
	pergeseran merah: CreateHsmConfiguration pergeseran merah: CreateIntegration pergeseran merah: CreateRedshiftIdcApplication pergeseran merah: CreateScheduledAction pergeseran merah: CreateSnapshotCopyGrant pergeseran merah: CreateSnapshotSchedule pergeseran merah: CreateUsageLimit pergeseran merah: DeauthorizeDataShare pergeseran merah: DeleteAuthenticationProfile pergeseran merah: DeleteCluster pergeseran merah: DeleteClusterParameterGroup pergeseran merah: DeleteClusterSecurityGroup pergeseran merah: DeleteClusterSnapshot pergeseran merah: DeleteClusterSubnetGroup pergeseran merah: DeleteCustomDomainAssociation pergeseran merah: DeleteEndpointAccess pergeseran merah: DeleteEventSubscription pergeseran merah: DeleteHsmClientCertificate pergeseran merah: DeleteHsmConfiguration pergeseran merah: DeletePartner pergeseran merah: DeleteRedshiftIdcApplication

Awalan layanan	Tindakan
	pergeseran merah: DeleteResourcePolicy
	pergeseran merah: DeleteScheduledAction
	pergeseran merah: DeleteSnapshotCopyGrant
	pergeseran merah: DeleteSnapshotSchedule
	pergeseran merah: DeleteUsageLimit
	pergeseran merah: DeregisterNamespace
	pergeseran merah: DescribeAccountAttributes
	pergeseran merah: DescribeAuthenticationProfiles
	pergeseran merah: DescribeClusterDbRevisions
	pergeseran merah: DescribeClusterParameterGroups
	pergeseran merah: DescribeClusterParameters
	pergeseran merah: DescribeClusterSecurityGroups
	pergeseran merah: DescribeClusterSnapshots
	pergeseran merah: DescribeClusterSubnetGroups
	pergeseran merah: DescribeClusterTracks
	pergeseran merah: DescribeClusterVersions
	pergeseran merah: DescribeClusters
	pergeseran merah: DescribeCustomDomainAssociations
	pergeseran merah: DescribeDataShares
	pergeseran merah: DescribeDataSharesForConsumer
	pergeseran merah: DescribeDataSharesForProducer

Awalan layanan	Tindakan
	pergeseran merah: DescribeDefaultClusterParameters
	pergeseran merah: DescribeEndpointAccess
	pergeseran merah: DescribeEndpointAuthorization
	pergeseran merah: DescribeEventCategories
	pergeseran merah: DescribeEventSubscriptions
	pergeseran merah: DescribeEvents
	pergeseran merah: DescribeHsmClientCertificates
	pergeseran merah: DescribeHsmConfigurations
	pergeseran merah: DescribeInboundIntegrations
	pergeseran merah: DescribeIntegrations
	pergeseran merah: DescribeLoggingStatus
	pergeseran merah: DescribeNodeConfigurationOptions
	pergeseran merah: DescribeOrderableClusterOptions
	pergeseran merah: DescribePartners
	pergeseran merah: DescribeRedshiftIdcApplications
	pergeseran merah: DescribeReservedNodeExchangeStatus
	pergeseran merah: DescribeReservedNodeOfferings
	pergeseran merah: DescribeReservedNodes
	pergeseran merah: DescribeResize
	pergeseran merah: DescribeScheduledActions
	pergeseran merah: DescribeSnapshotCopyGrants

Awalan layanan	Tindakan
	pergeseran merah: DescribeSnapshotSchedules pergeseran merah: DescribeStorage pergeseran merah: DescribeTableRestoreStatus pergeseran merah: DescribeUsageLimits pergeseran merah: DisableLogging pergeseran merah: DisableSnapshotCopy pergeseran merah: DisassociateDataShareConsumer pergeseran merah: EnableLogging pergeseran merah: EnableSnapshotCopy pergeseran merah: FailoverPrimaryCompute pergeseran merah: GetClusterCredentials pergeseran merah: GetClusterCredentialsWithIAM pergeseran merah: GetIdentityCenterAuthToken pergeseran merah: GetReservedNodeExchangeConfiguration Options pergeseran merah: GetReservedNodeExchangeOfferings pergeseran merah: GetResourcePolicy pergeseran merah: ListRecommendations pergeseran merah: ModifyAquaConfiguration pergeseran merah: ModifyAuthenticationProfile pergeseran merah: ModifyCluster

Awalan layanan	Tindakan
	pergeseran merah: ModifyClusterDbRevision pergeseran merah: ModifyClusterIamRoles pergeseran merah: ModifyClusterMaintenance pergeseran merah: ModifyClusterParameterGroup pergeseran merah: ModifyClusterSnapshot pergeseran merah: ModifyClusterSnapshotSchedule pergeseran merah: ModifyClusterSubnetGroup pergeseran merah: ModifyCustomDomainAssociation pergeseran merah: ModifyEndpointAccess pergeseran merah: ModifyEventSubscription pergeseran merah: ModifyRedshiftIdcApplication pergeseran merah: ModifyScheduledAction pergeseran merah: ModifySnapshotCopyRetentionPeriod pergeseran merah: ModifySnapshotSchedule pergeseran merah: ModifyUsageLimit pergeseran merah: PauseCluster pergeseran merah: PurchaseReservedNodeOffering pergeseran merah: PutResourcePolicy pergeseran merah: RebootCluster pergeseran merah: RegisterNamespace pergeseran merah: RejectDataShare

Awalan layanan	Tindakan
	pergeseran merah: ResetClusterParameterGroup pergeseran merah: ResizeCluster pergeseran merah: RestoreFromClusterSnapshot pergeseran merah: RestoreTableFromClusterSnapshot pergeseran merah: ResumeCluster pergeseran merah: RevokeClusterSecurityGroupIngress pergeseran merah: RevokeEndpointAccess pergeseran merah: RevokeSnapshotAccess pergeseran merah: RotateEncryptionKey pergeseran merah: UpdatePartnerStatus
data pergeseran merah	data pergeseran merah: BatchExecuteStatement data pergeseran merah: CancelStatement data pergeseran merah: DescribeStatement data pergeseran merah: DescribeTable data pergeseran merah: ExecuteStatement data pergeseran merah: GetStatementResult data pergeseran merah: ListDatabases data pergeseran merah: ListSchemas data pergeseran merah: ListSessions data pergeseran merah: ListStatements data pergeseran merah: ListTables

Awalan layanan	Tindakan
ruang refaktor	ruang refaktor: CreateApplication ruang refaktor: CreateEnvironment ruang refaktor: CreateRoute ruang refaktor: CreateService ruang refaktor: DeleteApplication ruang refaktor: DeleteEnvironment ruang refaktor: DeleteResourcePolicy ruang refaktor: DeleteRoute ruang refaktor: DeleteService ruang refaktor: GetApplication ruang refaktor: GetEnvironment ruang refaktor: GetResourcePolicy ruang refaktor: GetRoute ruang refaktor: GetService ruang refaktor: ListApplications ruang refaktor: ListEnvironmentVpcs ruang refaktor: ListEnvironments ruang refaktor: ListRoutes ruang refaktor: ListServices ruang refaktor: PutResourcePolicy ruang refaktor: UpdateRoute

Awalan layanan	Tindakan
rekognisi	Rekognisi: AssociateFaces
	Rekognisi: CompareFaces
	Rekognisi: CopyProjectVersion
	Rekognisi: CreateCollection
	Rekognisi: CreateDataset
	Rekognisi: CreateFaceLivenessSession
	Rekognisi: CreateProject
	Rekognisi: CreateProjectVersion
	Rekognisi: CreateStreamProcessor
	Rekognisi: CreateUser
	Rekognisi: DeleteCollection
	Rekognisi: DeleteDataset
	Rekognisi: DeleteFaces
	Rekognisi: DeleteProject
	Rekognisi: DeleteProjectPolicy
	Rekognisi: DeleteProjectVersion
	Rekognisi: DeleteStreamProcessor
	Rekognisi: DeleteUser
	Rekognisi: DescribeCollection
	Rekognisi: DescribeDataset
	Rekognisi: DescribeProjectVersions

Awalan layanan	Tindakan
	Rekognisi: DescribeProjects
	Rekognisi: DescribeStreamProcessor
	Rekognisi: DetectCustomLabels
	Rekognisi: DetectFaces
	Rekognisi: DetectLabels
	Rekognisi: DetectModerationLabels
	Rekognisi: DetectProtectiveEquipment
	Rekognisi: DetectText
	Rekognisi: DisassociateFaces
	Rekognisi: DistributeDatasetEntries
	Rekognisi: GetCelebrityInfo
	Rekognisi: GetCelebrityRecognition
	Rekognisi: GetContentModeration
	Rekognisi: GetFaceDetection
	Rekognisi: GetFaceLivenessSessionResults
	Rekognisi: GetFaceSearch
	Rekognisi: GetLabelDetection
	Rekognisi: GetMediaAnalysisJob
	Rekognisi: GetPersonTracking
	Rekognisi: GetSegmentDetection
	Rekognisi: GetTextDetection

Awalan layanan	Tindakan
	Rekognisi: IndexFaces
	Rekognisi: ListCollections
	Rekognisi: ListDatasetEntries
	Rekognisi: ListDatasetLabels
	Rekognisi: ListFaces
	Rekognisi: ListMediaAnalysisJobs
	Rekognisi: ListProjectPolicies
	Rekognisi: ListStreamProcessors
	Rekognisi: ListUsers
	Rekognisi: PutProjectPolicy
	Rekognisi: RecognizeCelebrities
	Rekognisi: SearchFaces
	Rekognisi: SearchFacesByImage
	Rekognisi: SearchUsers
	Rekognisi: SearchUsersByImage
	Rekognisi: StartCelebrityRecognition
	Rekognisi: StartContentModeration
	Rekognisi: StartFaceDetection
	Rekognisi: StartFaceLivenessSession
	Rekognisi: StartFaceSearch
	Rekognisi: StartLabelDetection

Awalan layanan	Tindakan
	Rekognisi: StartMediaAnalysisJob
	Rekognisi: StartPersonTracking
	Rekognisi: StartProjectVersion
	Rekognisi: StartSegmentDetection
	Rekognisi: StartStreamProcessor
	Rekognisi: StartTextDetection
	Rekognisi: StopProjectVersion
	Rekognisi: StopStreamProcessor
	Rekognisi: UpdateDatasetEntries
	Rekognisi: UpdateStreamProcessor

Awalan layanan	Tindakan
resiliencehub	resiliencehub: AcceptResourceGroupingRecommendations
	resiliencehub: AddDraftAppVersionResourceMappings
	resiliencehub: BatchUpdateRecommendationStatus
	resiliencehub: CreateApp
	resiliencehub: CreateAppVersionAppComponent
	resiliencehub: CreateAppVersionResource
	resiliencehub: CreateRecommendationTemplate
	resiliencehub: CreateResiliencyPolicy
	resiliencehub: DeleteApp
	resiliencehub: DeleteAppAssessment
	resiliencehub: DeleteAppInputSource
	resiliencehub: DeleteAppVersionAppComponent
	resiliencehub: DeleteAppVersionResource
	resiliencehub: DeleteRecommendationTemplate
	resiliencehub: DeleteResiliencyPolicy
	resiliencehub: DescribeApp
	resiliencehub: DescribeAppAssessment
	resiliencehub: DescribeAppVersion
	resiliencehub: DescribeAppVersionAppComponent
	resiliencehub: DescribeAppVersionResource
	resiliencehub: DescribeAppVersionResourcesResolutionStatus

Awalan layanan	Tindakan
	resiliencehub: DescribeAppVersionTemplate resiliencehub: DescribeDraftAppVersionResourcesImportStatus resiliencehub: DescribeMetricsExport resiliencehub: DescribeResiliencyPolicy resiliencehub: DescribeResourceGroupingRecommendationTask resiliencehub: ImportResourcesToDraftAppVersion resiliencehub: ListAlarmRecommendations resiliencehub: ListAppAssessmentComplianceDrifts resiliencehub: ListAppAssessmentResourceDrifts resiliencehub: ListAppAssessments resiliencehub: ListAppComponentCompliances resiliencehub: ListAppComponentRecommendations resiliencehub: ListAppInputSources resiliencehub: ListAppVersionAppComponents resiliencehub: ListAppVersionResourceMappings resiliencehub: ListAppVersionResources resiliencehub: ListAppVersions resiliencehub: ListApps resiliencehub: ListMetrics resiliencehub: ListRecommendationTemplates resiliencehub: ListResiliencyPolicies

Awalan layanan	Tindakan
	resiliencehub: ListResourceGroupingRecommendations resiliencehub: ListSopRecommendations resiliencehub: ListSuggestedResiliencyPolicies resiliencehub: ListTestRecommendations resiliencehub: ListUnsupportedAppVersionResources resiliencehub: PublishAppVersion resiliencehub: PutDraftAppVersionTemplate resiliencehub: RejectResourceGroupingRecommendations resiliencehub: RemoveDraftAppVersionResourceMappings resiliencehub: ResolveAppVersionResources resiliencehub: StartAppAssessment resiliencehub: StartResourceGroupingRecommendationTask resiliencehub: UpdateApp resiliencehub: UpdateAppVersion resiliencehub: UpdateAppVersionAppComponent resiliencehub: UpdateAppVersionResource resiliencehub: UpdateResiliencyPolicy

Awalan layanan	Tindakan
sumber daya-penjelajah-2	sumber daya-penjelajah-2: AssociateDefaultView
	sumber daya-penjelajah-2: BatchGetView
	sumber daya-penjelajah-2: CreateIndex
	sumber daya-penjelajah-2: CreateResourceExplorerSetup
	sumber daya-penjelajah-2: CreateView
	sumber daya-penjelajah-2: DeleteIndex
	sumber daya-penjelajah-2: DeleteResourceExplorerSetup
	sumber daya-penjelajah-2: DeleteView
	sumber daya-penjelajah-2: DisassociateDefaultView
	sumber daya-penjelajah-2: GetAccountLevelServiceConfiguration
	sumber daya-penjelajah-2: GetDefaultView
	sumber daya-penjelajah-2: GetIndex
	sumber daya-penjelajah-2: GetManagedView
	sumber daya-penjelajah-2: GetResourceExplorerSetup
	sumber daya-penjelajah-2: GetServiceIndex
	sumber daya-penjelajah-2: GetServiceView
	sumber daya-penjelajah-2: ListIndexes
	sumber daya-penjelajah-2: ListIndexesForMembers
	sumber daya-penjelajah-2: ListManagedViews
	sumber daya-penjelajah-2: ListServiceIndexes
	sumber daya-penjelajah-2: ListServiceViews

Awalan layanan	Tindakan
	sumber daya-penjelajah-2: ListStreamingAccessForServices
	sumber daya-penjelajah-2: ListSupportedResourceTypes
	sumber daya-penjelajah-2: ListViews
	Resource-Explorer-2:Search
	sumber daya-penjelajah-2: UpdateIndexType
	sumber daya-penjelajah-2: UpdateView

Awalan layanan	Tindakan
kelompok sumber daya	kelompok sumber daya: CancelTagSyncTask kelompok sumber daya: GetAccountSettings kelompok sumber daya: GetGroup kelompok sumber daya: GetGroupConfiguration kelompok sumber daya: GetGroupQuery kelompok sumber daya: GetTagSyncTask kelompok sumber daya: GroupResources kelompok sumber daya: ListGroupResources kelompok sumber daya: ListGroupingStatuses kelompok sumber daya: ListGroups kelompok sumber daya: ListTagSyncTasks kelompok sumber daya: PutGroupConfiguration kelompok sumber daya: SearchResources kelompok sumber daya: StartTagSyncTask kelompok sumber daya: UngroupResources kelompok sumber daya: UpdateAccountSettings kelompok sumber daya: UpdateGroup kelompok sumber daya: UpdateGroupQuery

Awalan layanan	Tindakan
robomaker	pembuat robomak: BatchDeleteWorlds
	pembuat robomak: BatchDescribeSimulationJob
	pembuat robomak: CancelDeploymentJob
	pembuat robomak: CancelSimulationJob
	pembuat robomak: CancelSimulationJobBatch
	pembuat robomak: CancelWorldExportJob
	pembuat robomak: CancelWorldGenerationJob
	pembuat robomak: CreateDeploymentJob
	pembuat robomak: CreateFleet
	pembuat robomak: CreateRobot
	pembuat robomak: CreateRobotApplication
	pembuat robomak: CreateRobotApplicationVersion
	pembuat robomak: CreateSimulationApplication
	pembuat robomak: CreateSimulationApplicationVersion
	pembuat robomak: CreateSimulationJob
	pembuat robomak: CreateWorldExportJob
	pembuat robomak: CreateWorldGenerationJob
	pembuat robomak: CreateWorldTemplate
	pembuat robomak: DeleteFleet
	pembuat robomak: DeleteRobot
	pembuat robomak: DeleteRobotApplication

Awalan layanan	Tindakan
	pembuat robomak: DeleteSimulationApplication
	pembuat robomak: DeleteWorldTemplate
	pembuat robomak: DeregisterRobot
	pembuat robomak: DescribeDeploymentJob
	pembuat robomak: DescribeFleet
	pembuat robomak: DescribeRobot
	pembuat robomak: DescribeRobotApplication
	pembuat robomak: DescribeSimulationApplication
	pembuat robomak: DescribeSimulationJob
	pembuat robomak: DescribeSimulationJobBatch
	pembuat robomak: DescribeWorld
	pembuat robomak: DescribeWorldExportJob
	pembuat robomak: DescribeWorldGenerationJob
	pembuat robomak: DescribeWorldTemplate
	pembuat robomak: GetWorldTemplateBody
	pembuat robomak: ListDeploymentJobs
	pembuat robomak: ListFleets
	pembuat robomak: ListRobotApplications
	pembuat robomak: ListRobots
	pembuat robomak: ListSimulationApplications
	pembuat robomak: ListSimulationJobBatches

Awalan layanan	Tindakan
	pembuat robomak: ListSimulationJobs
	pembuat robomak: ListWorldExportJobs
	pembuat robomak: ListWorldGenerationJobs
	pembuat robomak: ListWorldTemplates
	pembuat robomak: ListWorlds
	pembuat robomak: RegisterRobot
	pembuat robomak: RestartSimulationJob
	pembuat robomak: StartSimulationJobBatch
	pembuat robomak: SyncDeploymentJob
	pembuat robomak: UpdateRobotApplication
	pembuat robomak: UpdateSimulationApplication
	pembuat robomak: UpdateWorldTemplate

Awalan layanan	Tindakan
peranandi mana saja	peranandi mana saja: CreateProfile peranandi mana saja: CreateTrustAnchor peranandi mana saja: DeleteAttributeMapping peranandi mana saja: DeleteCrl peranandi mana saja: DeleteProfile peranandi mana saja: DeleteTrustAnchor peranandi mana saja: DisableCrl peranandi mana saja: DisableProfile peranandi mana saja: DisableTrustAnchor peranandi mana saja: EnableCrl peranandi mana saja: EnableProfile peranandi mana saja: EnableTrustAnchor peranandi mana saja: GetCrl peranandi mana saja: GetProfile peranandi mana saja: GetSubject peranandi mana saja: GetTrustAnchor peranandi mana saja: ImportCrl peranandi mana saja: ListCrls peranandi mana saja: ListProfiles peranandi mana saja: ListSubjects peranandi mana saja: ListTrustAnchors

Awalan layanan	Tindakan
	peranandi mana saja: PutAttributeMapping peranandi mana saja: PutNotificationSettings peranandi mana saja: ResetNotificationSettings peranandi mana saja: UpdateCrl peranandi mana saja: UpdateProfile peranandi mana saja: UpdateTrustAnchor

Awalan layanan	Tindakan
route53	route53: ActivateKeySigningKey
	route53: AssociateVPCWithHostedZone
	route53: ChangeCidrCollection
	route53: ChangeResourceRecordSets
	route53: CreateCidrCollection
	route53: CreateHealthCheck
	route53: CreateHostedZone
	route53: CreateKeySigningKey
	route53: CreateQueryLoggingConfig
	route53: CreateReusableDelegationSet
	route53: CreateTrafficPolicy
	route53: CreateTrafficPolicyInstance
	route53: CreateTrafficPolicyVersion
	route53: CreateVPCAssociationAuthorization
	route53: DeactivateKeySigningKey
	route53: DeleteCidrCollection
	route53: DeleteHealthCheck
	route53: DeleteHostedZone
	route53: DeleteKeySigningKey
	route53: DeleteQueryLoggingConfig
	route53: DeleteReusableDelegationSet

Awalan layanan	Tindakan
	route53: DeleteTrafficPolicy
	route53: DeleteTrafficPolicyInstance
	route53: DeleteVPCAssociationAuthorization
	route53: DisableHostedZoneDNSSEC
	route53: DisassociateVPCFromHostedZone
	route53: EnableHostedZoneDNSSEC
	route53: GetAccountLimit
	route53: GetChange
	route53: GetCheckerIpRanges
	Route53:GetDNSSec
	route53: GetGeoLocation
	route53: GetHealthCheck
	route53: GetHealthCheckCount
	route53: GetHealthCheckLastFailureReason
	route53: GetHealthCheckStatus
	route53: GetHostedZone
	route53: GetHostedZoneCount
	route53: GetHostedZoneLimit
	route53: GetQueryLoggingConfig
	route53: GetReusableDelegationSet
	route53: GetReusableDelegationSetLimit

Awalan layanan	Tindakan
	route53: GetTrafficPolicy
	route53: GetTrafficPolicyInstance
	route53: GetTrafficPolicyInstanceCount
	route53: ListCidrBlocks
	route53: ListCidrCollections
	route53: ListCidrLocations
	route53: ListGeoLocations
	route53: ListHealthChecks
	route53: ListHostedZones
	route53: ListHostedZonesByName
	route53: ListHostedZonesByVPC
	route53: ListQueryLoggingConfigs
	route53: ListResourceRecordSets
	route53: ListReusableDelegationSets
	route53: ListTrafficPolicies
	route53: ListTrafficPolicyInstances
	route53: ListTrafficPolicyInstancesByHostedZone
	route53: ListTrafficPolicyInstancesByPolicy
	route53: ListTrafficPolicyVersions
	route53: ListVPCAssociationAuthorizations
	Route53:testDNSsAnswer

Awalan layanan	Tindakan
	route53: UpdateHealthCheck
	route53: UpdateHostedZoneComment
	route53: UpdateTrafficPolicyComment
	route53: UpdateTrafficPolicyInstance

Awalan layanan	Tindakan
route53-recovery-control-config	route53-recovery-control-config: CreateCluster route53-recovery-control-config: CreateControlPanel route53-recovery-control-config: CreateRoutingControl route53-recovery-control-config: CreateSafetyRule route53-recovery-control-config: DeleteCluster route53-recovery-control-config: DeleteControlPanel route53-recovery-control-config: DeleteRoutingControl route53-recovery-control-config: DeleteSafetyRule route53-recovery-control-config: DescribeCluster route53-recovery-control-config: DescribeControlPanel route53-recovery-control-config: DescribeRoutingControl route53-recovery-control-config: DescribeSafetyRule route53-recovery-control-config: GetResourcePolicy route53-recovery-control-config: ListAssociatedRoute53HealthChecks route53-recovery-control-config: ListClusters route53-recovery-control-config: ListControlPanels route53-recovery-control-config: ListRoutingControls route53-recovery-control-config: ListSafetyRules route53-recovery-control-config: UpdateCluster route53-recovery-control-config: UpdateControlPanel

Awalan layanan	Tindakan
	route53-recovery-control-config: UpdateRoutingControl route53-recovery-control-config: UpdateSafetyRule

Awalan layanan	Tindakan
route53-pemulihan-kesiapan	route53-pemulihan-kesiapan: CreateCell route53-pemulihan-kesiapan: CreateCrossAccountAuthorization route53-pemulihan-kesiapan: CreateReadinessCheck route53-pemulihan-kesiapan: CreateRecoveryGroup route53-pemulihan-kesiapan: CreateResourceSet route53-pemulihan-kesiapan: DeleteCell route53-pemulihan-kesiapan: DeleteCrossAccountAuthorization route53-pemulihan-kesiapan: DeleteReadinessCheck route53-pemulihan-kesiapan: DeleteRecoveryGroup route53-pemulihan-kesiapan: DeleteResourceSet route53-pemulihan-kesiapan: GetArchitectureRecommendations route53-pemulihan-kesiapan: GetCell route53-pemulihan-kesiapan: GetCellReadinessSummary route53-pemulihan-kesiapan: GetReadinessCheck route53-pemulihan-kesiapan: GetReadinessCheckResourceStatus route53-pemulihan-kesiapan: GetReadinessCheckStatus route53-pemulihan-kesiapan: GetRecoveryGroup route53-pemulihan-kesiapan: GetRecoveryGroupReadinessSummary route53-pemulihan-kesiapan: GetResourceSet route53-pemulihan-kesiapan: ListCells

Awalan layanan	Tindakan
	route53-pemulihan-kesiapan: ListCrossAccountAuthorizations
	route53-pemulihan-kesiapan: ListReadinessChecks
	route53-pemulihan-kesiapan: ListRecoveryGroups
	route53-pemulihan-kesiapan: ListResourceSets
	route53-pemulihan-kesiapan: ListRules
	route53-pemulihan-kesiapan: UpdateCell
	route53-pemulihan-kesiapan: UpdateReadinessCheck
	route53-pemulihan-kesiapan: UpdateRecoveryGroup
	route53-pemulihan-kesiapan: UpdateResourceSet

Awalan layanan	Tindakan
route53resolver	route53resolver: AssociateFirewallRuleGroup route53resolver: AssociateResolverEndpointIpAddress route53resolver: AssociateResolverQueryLogConfig route53resolver: AssociateResolverRule route53resolver: CreateFirewallDomainList route53resolver: CreateFirewallRule route53resolver: CreateFirewallRuleGroup route53resolver: CreateResolverEndpoint route53resolver: CreateResolverQueryLogConfig route53resolver: CreateResolverRule route53resolver: DeleteFirewallDomainList route53resolver: DeleteFirewallRule route53resolver: DeleteFirewallRuleGroup route53resolver: DeleteOutpostResolver route53resolver: DeleteResolverEndpoint route53resolver: DeleteResolverQueryLogConfig route53resolver: DeleteResolverRule route53resolver: DisassociateFirewallRuleGroup route53resolver: DisassociateResolverEndpointIpAddress route53resolver: DisassociateResolverQueryLogConfig route53resolver: DisassociateResolverRule

Awalan layanan	Tindakan
	route53resolver: GetFirewallConfig
	route53resolver: GetFirewallDomainList
	route53resolver: GetFirewallRuleGroup
	route53resolver: GetFirewallRuleGroupAssociation
	route53resolver: GetFirewallRuleGroupPolicy
	route53resolver: GetOutpostResolver
	route53resolver: GetResolverConfig
	route53resolver: GetResolverDnssecConfig
	route53resolver: GetResolverEndpoint
	route53resolver: GetResolverQueryLogConfig
	route53resolver: GetResolverQueryLogConfigAssociation
	route53resolver: GetResolverQueryLogConfigPolicy
	route53resolver: GetResolverRule
	route53resolver: GetResolverRuleAssociation
	route53resolver: GetResolverRulePolicy
	route53resolver: ImportFirewallDomains
	route53resolver: ListFirewallConfigs
	route53resolver: ListFirewallDomainLists
	route53resolver: ListFirewallDomains
	route53resolver: ListFirewallRuleGroupAssociations
	route53resolver: ListFirewallRuleGroups

Awalan layanan	Tindakan
	route53resolver: ListFirewallRuleTypes route53resolver: ListFirewallRules route53resolver: ListOutpostResolvers route53resolver: ListResolverConfigs route53resolver: ListResolverDnssecConfigs route53resolver: ListResolverEndpointIpAddresses route53resolver: ListResolverEndpoints route53resolver: ListResolverQueryLogConfigAssociations route53resolver: ListResolverQueryLogConfigs route53resolver: ListResolverRuleAssociations route53resolver: ListResolverRules route53resolver: PutFirewallRuleGroupPolicy route53resolver: PutResolverQueryLogConfigPolicy route53resolver: UpdateFirewallConfig route53resolver: UpdateFirewallDomains route53resolver: UpdateFirewallRule route53resolver: UpdateFirewallRuleGroupAssociation route53resolver: UpdateOutpostResolver route53resolver: UpdateResolverConfig route53resolver: UpdateResolverDnssecConfig route53resolver: UpdateResolverEndpoint

Awalan layanan	Tindakan
	route53resolver: UpdateResolverRule
rum	rum: BatchCreateRumMetricDefinitions
	rum: BatchDeleteRumMetricDefinitions
	rum: BatchGetRumMetricDefinitions
	rum: CreateAppMonitor
	rum: DeleteAppMonitor
	rum: DeleteResourcePolicy
	rum: DeleteRumMetricsDestination
	rum: GetAppMonitor
	rum: GetAppMonitorData
	rum: GetResourcePolicy
	rum: ListAppMonitors
	rum: ListRumMetricsDestinations
	rum: PutResourcePolicy
	rum: PutRumMetricsDestination
	rum: UpdateAppMonitor
	rum: UpdateRumMetricDefinition

Awalan layanan	Tindakan
s3	s3: AssociateAccessGrantsIdentityCenter s3: CreateAccessGrant s3: CreateAccessGrantsInstance s3: CreateAccessGrantsLocation s3: CreateAccessPoint s3: CreateAccessPointForObjectLambda s3: CreateBucket s3: CreateBucketMetadataTableConfiguration s3: CreateJob s3: CreateMultiRegionAccessPoint s3: DeleteAccessGrant s3: DeleteAccessGrantsInstance s3: DeleteAccessGrantsInstanceResourcePolicy s3: DeleteAccessGrantsLocation s3: DeleteAccessPoint s3: DeleteAccessPointForObjectLambda s3: DeleteAccessPointPolicy s3: DeleteAccessPointPolicyForObjectLambda s3: DeleteBucket s3: DeleteBucketMetadataTableConfiguration s3: DeleteBucketPolicy

Awalan layanan	Tindakan
	s3: DeleteBucketWebsite s3: DeleteMultiRegionAccessPoint s3: DeleteStorageLensConfiguration s3: DescribeJob s3: DescribeMultiRegionAccessPointOperation s3: DissociateAccessGrantsIdentityCenter s3: GetAccelerateConfiguration s3: GetAccessGrant s3: GetAccessGrantsInstance s3: GetAccessGrantsInstanceForPrefix s3: GetAccessGrantsInstanceResourcePolicy s3: GetAccessGrantsLocation s3: GetAccessPoint s3: GetAccessPointConfigurationForObjectLambda s3: GetAccessPointForObjectLambda s3: GetAccessPointPolicy s3: GetAccessPointPolicyForObjectLambda s3: GetAccessPointPolicyStatus s3: GetAccessPointPolicyStatusForObjectLambda s3: GetAccountPublicAccessBlock s3: GetAnalyticsConfiguration

Awalan layanan	Tindakan
	s3: GetBucketAbac s3: GetBucketAcl s3: GetBucket CORS s3: GetBucketLocation s3: GetBucketLogging s3: GetBucketNotification s3: GetBucketObjectLockConfiguration s3: GetBucketOwnershipControls s3: GetBucketPolicy s3: GetBucketPolicyStatus s3: GetBucketPublicAccessBlock s3: GetBucketRequestPayment s3: GetBucketVersioning s3: GetBucketWebsite s3: GetDataAccess s3: GetEncryptionConfiguration s3: GetIntelligentTieringConfiguration s3: GetInventoryConfiguration s3: GetLifecycleConfiguration s3: GetMetricsConfiguration s3: GetMultiRegionAccessPoint

Awalan layanan	Tindakan
	s3: GetMultiRegionAccessPointPolicy s3: GetMultiRegionAccessPointPolicyStatus s3: GetMultiRegionAccessPointRoutes s3: GetReplicationConfiguration s3: GetStorageLensConfiguration s3: GetStorageLensDashboard s3: ListAccessGrants s3: ListAccessGrantsInstances s3: ListAccessGrantsLocations s3: ListAccessPoints s3: ListAccessPointsForObjectLambda s3: ListAllMyBuckets s3: ListBucketMultipartUploads s3: ListCallerAccessGrants s3: ListJobs s3: ListMultiRegionAccessPoints s3: ListStorageLensConfigurations s3: PutAccelerateConfiguration s3: PutAccessGrantsInstanceResourcePolicy s3: PutAccessPointConfigurationForObjectLambda s3: PutAccessPointPolicy

Awalan layanan	Tindakan
	s3: PutAccessPointPolicyForObjectLambda s3: PutAccountPublicAccessBlock s3: PutAnalyticsConfiguration s3: PutBucketAbac s3: PutBucketAcl s3: PutBucket CORS s3: PutBucketLogging s3: PutBucketNotification s3: PutBucketObjectLockConfiguration s3: PutBucketOwnershipControls s3: PutBucketPolicy s3: PutBucketPublicAccessBlock s3: PutBucketRequestPayment s3: PutBucketVersioning s3: PutBucketWebsite s3: PutEncryptionConfiguration s3: PutIntelligentTieringConfiguration s3: PutInventoryConfiguration s3: PutLifecycleConfiguration s3: PutMetricsConfiguration s3: PutMultiRegionAccessPointPolicy

Awalan layanan	Tindakan
	s3: PutReplicationConfiguration s3: PutStorageLensConfiguration s3: SubmitMultiRegionAccessPointRoutes s3: UpdateAccessGrantsLocation s3: UpdateBucketMetadataAnnotationTableConfiguration s3: UpdateBucketMetadataJournalTableConfiguration s3: UpdateJobPriority s3: UpdateJobStatus
pos terdepan s3	pos terdepan s3: CreateEndpoint pos terdepan s3: DeleteEndpoint pos terdepan s3: ListEndpoints pos terdepan s3: ListOutpostsWithS3 pos terdepan s3: ListSharedEndpoints

Awalan layanan	Tindakan
sagemaker-geospasial	sagemaker-geospasial: DeleteEarthObservationJob sagemaker-geospasial: DeleteVectorEnrichmentJob sagemaker-geospasial: ExportEarthObservationJob sagemaker-geospasial: ExportVectorEnrichmentJob sagemaker-geospasial: GetEarthObservationJob sagemaker-geospasial: GetRasterDataCollection sagemaker-geospasial: GetTile sagemaker-geospasial: GetVectorEnrichmentJob sagemaker-geospasial: ListEarthObservationJobs sagemaker-geospasial: ListRasterDataCollections sagemaker-geospasial: ListVectorEnrichmentJobs sagemaker-geospasial: SearchRasterDataCollection sagemaker-geospasial: StartEarthObservationJob sagemaker-geospasial: StartVectorEnrichmentJob sagemaker-geospasial: StopEarthObservationJob sagemaker-geospasial: StopVectorEnrichmentJob

Awalan layanan	Tindakan
savingsplans	savingsplans: CreateSavingsPlan
	savingsplans: DeleteQueuedSavingsPlan
	savingsplans: DescribeSavingsPlanRates
	savingsplans: DescribeSavingsPlans
	savingsplans: DescribeSavingsPlansOfferingRates
	savingsplans: DescribeSavingsPlansOfferings
	savingsplans: ReturnSavingsPlan

Awalan layanan	Tindakan
skema	skema: CreateDiscoverer skema: CreateRegistry skema: CreateSchema skema: DeleteDiscoverer skema: DeleteRegistry skema: DeleteResourcePolicy skema: DeleteSchema skema: DeleteSchemaVersion skema: DescribeCodeBinding skema: DescribeDiscoverer skema: DescribeRegistry skema: DescribeSchema skema: ExportSchema skema: GetCodeBindingSource skema: GetDiscoveredSchema skema: GetResourcePolicy skema: ListDiscoverers skema: ListRegistries skema: ListSchemaVersions skema: ListSchemas skema: PutCodeBinding

Awalan layanan	Tindakan
	skema: PutResourcePolicy skema: SearchSchemas skema: StartDiscoverer skema: StopDiscoverer skema: UpdateDiscoverer skema: UpdateRegistry skema: UpdateSchema
sdb	sdb: CreateDomain sdb: DeleteDomain sdb: DomainMetadata sdb: ListDomains

Awalan layanan	Tindakan
manajer rahasia	manajer rahasia: CancelRotateSecret manajer rahasia: CreateSecret manajer rahasia: DeleteResourcePolicy manajer rahasia: DeleteSecret manajer rahasia: DescribeSecret manajer rahasia: GetRandomPassword manajer rahasia: GetResourcePolicy manajer rahasia: GetSecretValue manajer rahasia: ListSecretVersionIds manajer rahasia: ListSecrets manajer rahasia: PutResourcePolicy manajer rahasia: PutSecretValue manajer rahasia: RemoveRegionsFromReplication manajer rahasia: ReplicateSecretToRegions manajer rahasia: RestoreSecret manajer rahasia: RotateSecret manajer rahasia: StopReplicationToReplica manajer rahasia: UpdateSecret manajer rahasia: ValidateResourcePolicy

Awalan layanan	Tindakan
securityhub	securityhub: AcceptAdministratorInvitation
	securityhub: AcceptInvitation
	securityhub: BatchDeleteAutomationRules
	securityhub: BatchDisableStandards
	securityhub: BatchEnableStandards
	securityhub: BatchGetAutomationRules
	securityhub: BatchGetConfigurationPolicyAssociations
	securityhub: BatchGetEnabledRegionsV2
	securityhub: BatchGetSecurityControls
	securityhub: BatchGetStandardsControlAssociations
	securityhub: BatchImportFindings
	securityhub: BatchUpdateAutomationRules
	securityhub: BatchUpdateFindings
	securityhub: BatchUpdateStandardsControlAssociations
	securityhub: V2 ConnectorRegistrations
	securityhub: CreateActionTarget
	securityhub: V2 CreateAggregator
	securityhub: CreateAutomationRule
	securityhub: CreateAutomationRuleV2
	securityhub: CreateConfigurationPolicy
	securityhub: CreateConnector

Awalan layanan	Tindakan
	securityhub: V2 CreateConnector
	securityhub: CreateFindingAggregator
	securityhub: CreateInsight
	securityhub: CreateMembers
	securityhub: V2 CreateTicket
	securityhub: DeclineInvitations
	securityhub: DeleteActionTarget
	securityhub: V2 DeleteAggregator
	securityhub: DeleteAutomationRuleV2
	securityhub: DeleteConfigurationPolicy
	securityhub: DeleteConnector
	securityhub: V2 DeleteConnector
	securityhub: DeleteFindingAggregator
	securityhub: DeleteInsight
	securityhub: DeleteInvitations
	securityhub: DeleteMembers
	securityhub: DescribeActionTargets
	securityhub: DescribeHub
	securityhub: DescribeOrganizationConfiguration
	securityhub: DescribeProducts
	securityhub: DescribeSecurityHubV2

Awalan layanan	Tindakan
	securityhub: DescribeStandards
	securityhub: DisableImportFindingsForProduct
	securityhub: DisableOrganizationAdminAccount
	securityhub: DisableSecurityHub
	securityhub: DisableSecurityHubFeatureV2
	securityhub: DisableSecurityHubV2
	securityhub: DisassociateFromAdministratorAccount
	securityhub: DisassociateFromMasterAccount
	securityhub: DisassociateMembers
	securityhub: EnableImportFindingsForProduct
	securityhub: EnableOrganizationAdminAccount
	securityhub: EnableSecurityHub
	securityhub: EnableSecurityHubFeatureV2
	securityhub: GetAdministratorAccount
	securityhub: V2 GetAggregator
	securityhub: GetAutomationRuleV2
	securityhub: GetConfigurationPolicy
	securityhub: GetConfigurationPolicyAssociation
	securityhub: GetConnector
	securityhub: V2 GetConnector
	securityhub: GetCoverageStatisticsV2

Awalan layanan	Tindakan
	securityhub: GetEnabledStandards
	securityhub: GetFindingAggregator
	securityhub: GetFindingHistory
	securityhub: GetFindings
	securityhub: GetInsightResults
	securityhub: GetInsights
	securityhub: GetInvitationsCount
	securityhub: GetMasterAccount
	securityhub: GetMembers
	securityhub: GetSecurityControlDefinition
	securityhub: InviteMembers
	securityhub: V2 ListAggregators
	securityhub: ListAutomationRules
	securityhub: ListAutomationRulesV2
	securityhub: ListConfigurationPolicies
	securityhub: ListConfigurationPolicyAssociations
	securityhub: ListConnectors
	securityhub: V2 ListConnectors
	securityhub: ListEnabledProductsForImport
	securityhub: ListFindingAggregators
	securityhub: ListInvitations

Awalan layanan	Tindakan
	securityhub: ListMembers
	securityhub: ListOrganizationAdminAccounts
	securityhub: ListSecurityControlDefinitions
	securityhub: ListStandardsControlAssociations
	securityhub: StartConfigurationPolicyAssociation
	securityhub: StartConfigurationPolicyDisassociation
	securityhub: UpdateActionTarget
	securityhub: V2 UpdateAggregator
	securityhub: UpdateAutomationRuleV2
	securityhub: UpdateConfigurationPolicy
	securityhub: UpdateConnector
	securityhub: V2 UpdateConnector
	securityhub: UpdateFindingAggregator
	securityhub: UpdateFindings
	securityhub: UpdateInsight
	securityhub: UpdateOrganizationConfiguration
	securityhub: UpdateSecurityControl
	securityhub: UpdateSecurityHubConfiguration

Awalan layanan	Tindakan
Securitylake	Securitylake: CreateAwsLogSource Securitylake: CreateCustomLogSource Securitylake: CreateDataLakeExceptionSubscription Securitylake: CreateDataLakeOrganizationConfiguration Securitylake: CreateSubscriber Securitylake: CreateSubscriberNotification Securitylake: DeleteAwsLogSource Securitylake: DeleteCustomLogSource Securitylake: DeleteDataLakeExceptionSubscription Securitylake: DeleteDataLakeOrganizationConfiguration Securitylake: DeleteSubscriber Securitylake: DeleteSubscriberNotification Securitylake: DeregisterDataLakeDelegatedAdministrator Securitylake: GetDataLakeExceptionSubscription Securitylake: GetDataLakeOrganizationConfiguration Securitylake: GetDataLakeSources Securitylake: GetSubscriber Securitylake: ListDataLakes Securitylake: ListLogSources Securitylake: ListSubscribers Securitylake: RegisterDataLakeDelegatedAdministrator

Awalan layanan	Tindakan
	Securitylake: UpdateDataLakeExceptionSubscription Securitylake: UpdateSubscriber Securitylake: UpdateSubscriberNotification
repo tanpa server	repo tanpa server: CreateApplication repo tanpa server: CreateApplicationVersion repo tanpa server: CreateCloudFormationChangeSet repo tanpa server: CreateCloudFormationTemplate repo tanpa server: DeleteApplication repo tanpa server: GetApplication repo tanpa server: GetApplicationPolicy repo tanpa server: GetCloudFormationTemplate repo tanpa server: ListApplicationDependencies repo tanpa server: ListApplicationVersions repo tanpa server: ListApplications repo tanpa server: PutApplicationPolicy repo tanpa server: UnshareApplication repo tanpa server: UpdateApplication

Awalan layanan	Tindakan
servicecatalog	servicecatalog: AcceptPortfolioShare servicecatalog: AssociateBudgetWithResource servicecatalog: AssociatePrincipalWithPortfolio servicecatalog: AssociateProductWithPortfolio servicecatalog: AssociateServiceActionWithProvisioningArtifact servicecatalog: BatchAssociateServiceActionWithProvisioningArtifact servicecatalog: BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog: CopyProduct servicecatalog: CreateAttributeGroup servicecatalog: CreateConstraint servicecatalog: CreatePortfolio servicecatalog: CreatePortfolioShare servicecatalog: CreateProduct servicecatalog: CreateProvisionedProductPlan servicecatalog: CreateProvisioningArtifact servicecatalog: CreateServiceAction servicecatalog: DeleteAttributeGroup servicecatalog: DeleteConstraint servicecatalog: DeletePortfolio servicecatalog: DeletePortfolioShare

Awalan layanan	Tindakan
	servicecatalog: DeleteProduct servicecatalog: DeleteProvisionedProductPlan servicecatalog: DeleteProvisioningArtifact servicecatalog: DeleteServiceAction servicecatalog: DescribeConstraint servicecatalog: DescribeCopyProductStatus servicecatalog: DescribePortfolio servicecatalog: DescribePortfolioShareStatus servicecatalog: DescribePortfolioShares servicecatalog: DescribeProduct servicecatalog: DescribeProductAsAdmin servicecatalog: DescribeProductView servicecatalog: DescribeProvisionedProduct servicecatalog: DescribeProvisionedProductPlan servicecatalog: DescribeProvisioningArtifact servicecatalog: DescribeProvisioningParameters servicecatalog: DescribeRecord servicecatalog: DescribeServiceAction servicecatalog: DescribeServiceActionExecutionParameters servicecatalog: DisableAWSOrganizationsAccess servicecatalog: DisassociateBudgetFromResource

Awalan layanan	Tindakan
	servicecatalog: DisassociatePrincipalFromPortfolio
	servicecatalog: DisassociateProductFromPortfolio
	servicecatalog: DisassociateServiceActionFromProvisioningArtifact
	servicecatalog: EnableAWSOrganizationsAccess
	servicecatalog: ExecuteProvisionedProductPlan
	servicecatalog: ExecuteProvisionedProductServiceAction
	servicecatalog: GetAWSOrganizationsAccessStatus
	servicecatalog: GetProvisionedProductOutputs
	servicecatalog: ImportAsProvisionedProduct
	servicecatalog: ListAcceptedPortfolioShares
	servicecatalog: ListAttributeGroups
	servicecatalog: ListBudgetsForResource
	servicecatalog: ListConstraintsForPortfolio
	servicecatalog: ListLaunchPaths
	servicecatalog: ListOrganizationPortfolioAccess
	servicecatalog: ListPortfolioAccess
	servicecatalog: ListPortfolios
	servicecatalog: ListPortfoliosForProduct
	servicecatalog: ListPrincipalsForPortfolio
	servicecatalog: ListProvisionedProductPlans
	servicecatalog: ListProvisioningArtifacts

Awalan layanan	Tindakan
	servicecatalog: ListProvisioningArtifactsForServiceAction servicecatalog: ListRecordHistory servicecatalog: ListServiceActions servicecatalog: ListServiceActionsForProvisioningArtifact servicecatalog: ListStackInstancesForProvisionedProduct servicecatalog: NotifyProvisionProductEngineWorkflowResult servicecatalog: NotifyTerminateProvisionedProductEngineWorkflowResult servicecatalog: NotifyUpdateProvisionedProductEngineWorkflowResult servicecatalog: ProvisionProduct servicecatalog: RejectPortfolioShare servicecatalog: ScanProvisionedProducts servicecatalog: SearchProducts servicecatalog: SearchProductsAsAdmin servicecatalog: SearchProvisionedProducts servicecatalog: TerminateProvisionedProduct servicecatalog: UpdateConstraint servicecatalog: UpdatePortfolio servicecatalog: UpdatePortfolioShare servicecatalog: UpdateProduct servicecatalog: UpdateProvisionedProduct

Awalan layanan	Tindakan
	servicecatalog: UpdateProvisionedProductProperties servicecatalog: UpdateProvisioningArtifact servicecatalog: UpdateServiceAction

Awalan layanan	Tindakan
servicediscovery	penemuan layanan: CreateHttpNamespace
	penemuan layanan: CreatePrivateDnsNamespace
	penemuan layanan: CreatePublicDnsNamespace
	penemuan layanan: CreateService
	penemuan layanan: DeleteNamespace
	penemuan layanan: DeleteService
	penemuan layanan: DeleteServiceAttributes
	penemuan layanan: DeregisterInstance
	penemuan layanan: GetInstance
	penemuan layanan: GetInstancesHealthStatus
	penemuan layanan: GetNamespace
	penemuan layanan: GetOperation
	penemuan layanan: GetService
	penemuan layanan: ListInstances
	penemuan layanan: ListNamespaces
	penemuan layanan: ListOperations
	penemuan layanan: ListServices
	penemuan layanan: RegisterInstance
	penemuan layanan: UpdateHttpNamespace
	penemuan layanan: UpdateInstanceCustomHealthStatus
	penemuan layanan: UpdatePrivateDnsNamespace

Awalan layanan	Tindakan
	penemuan layanan: UpdatePublicDnsNamespace penemuan layanan: UpdateService penemuan layanan: UpdateServiceAttributes

Awalan layanan	Tindakan
servicequotas	servicequotas: AssociateServiceQuotaTemplate servicequotas: CreateSupportCase servicequotas: DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas: DisassociateServiceQuotaTemplate servicequotas: GetAWSDefaultServiceQuota servicequotas: GetAssociationForServiceQuotaTemplate servicequotas: GetAutoManagementConfiguration servicequotas: GetQuotaUtilizationReport servicequotas: GetRequestedServiceQuotaChange servicequotas: GetServiceQuota servicequotas: GetServiceQuotaIncreaseRequestFromTemplate servicequotas: ListAWSDefaultServiceQuotas servicequotas: ListRequestedServiceQuotaChangeHistory servicequotas: ListRequestedServiceQuotaChangeHistoryByQuota servicequotas: ListServiceQuotaIncreaseRequestsInTemplate servicequotas: ListServiceQuotas servicequotas: ListServices servicequotas: PutServiceQuotaIncreaseRequestIntoTemplate servicequotas: RequestServiceQuotaIncrease servicequotas: StartAutoManagement servicequotas: StartQuotaUtilizationReport

Awalan layanan	Tindakan
	servicequotas: StopAutoManagement servicequotas: UpdateAutoManagement

Awalan layanan	Tindakan
ses	ses: BatchGetMetricData
	ses: CloneReceiptRuleSet
	ses: CreateAddonInstance
	ses: CreateAddonSubscription
	ses: CreateAddressList
	ses: CreateAddressListImportJob
	ses: CreateArchive
	ses: CreateConfigurationSet
	ses: CreateConfigurationSetEventDestination
	ses: CreateConfigurationSetTrackingOptions
	ses: CreateContact
	ses: CreateContactList
	ses: CreateCustomVerificationEmailTemplate
	ses: CreateDedicatedIpPool
	ses: CreateDeliverabilityTestReport
	ses: CreateEmailIdentity
	ses: CreateEmailIdentityPolicy
	ses: CreateEmailTemplate
	ses: CreateImportJob
	ses: CreateIngressPoint
	ses: CreateMultiRegionEndpoint

Awalan layanan	Tindakan
	ses: CreateReceiptFilter
	ses: CreateReceiptRule
	ses: CreateReceiptRuleSet
	ses: CreateRelay
	ses: CreateRuleSet
	ses: CreateTemplate
	ses: CreateTenant
	ses: CreateTenantResourceAssociation
	ses: CreateTrafficPolicy
	ses: DeleteAddonInstance
	ses: DeleteAddonSubscription
	ses: DeleteAddressList
	ses: DeleteArchive
	ses: DeleteConfigurationSet
	ses: DeleteConfigurationSetEventDestination
	ses: DeleteConfigurationSetTrackingOptions
	ses: DeleteContact
	ses: DeleteContactList
	ses: DeleteCustomVerificationEmailTemplate
	ses: DeleteDedicatedIpPool
	ses: DeleteEmailIdentity

Awalan layanan	Tindakan
	ses: DeleteEmailIdentityPolicy
	ses: DeleteEmailTemplate
	ses: DeleteIdentity
	ses: DeleteIdentityPolicy
	ses: DeleteIngressPoint
	ses: DeleteMultiRegionEndpoint
	ses: DeleteReceiptFilter
	ses: DeleteReceiptRule
	ses: DeleteReceiptRuleSet
	ses: DeleteRelay
	ses: DeleteRuleSet
	ses: DeleteSuppressedDestination
	ses: DeleteTemplate
	ses: DeleteTenant
	ses: DeleteTenantResourceAssociation
	ses: DeleteTrafficPolicy
	ses: DeleteVerifiedEmailAddress
	ses: DeregisterMemberFromAddressList
	ses: DescribeActiveReceiptRuleSet
	ses: DescribeConfigurationSet
	ses: DescribeReceiptRule

Awalan layanan	Tindakan
	ses: DescribeReceiptRuleSet
	ses: GetAccount
	ses: GetAccountSendingEnabled
	ses: GetAddonInstance
	ses: GetAddonSubscription
	ses: GetAddressList
	ses: GetArchive
	ses: GetArchiveExport
	ses: GetArchiveMessage
	ses: GetArchiveMessageContent
	ses: GetArchiveSearch
	ses: GetArchiveSearchResults
	ses: GetBlacklistReports
	ses: GetConfigurationSet
	ses: GetConfigurationSetEventDestinations
	ses: GetContact
	ses: GetContactList
	ses: GetCustomVerificationEmailTemplate
	ses: GetDedicatedIp
	ses: GetDedicatedIpPool
	ses: GetDedicatedIps

Awalan layanan	Tindakan
	ses: GetDeliverabilityDashboardOptions
	ses: GetDeliverabilityTestReport
	ses: GetDomainDeliverabilityCampaign
	ses: GetDomainStatisticsReport
	ses: GetEmailAddressInsights
	ses: GetEmailIdentity
	ses: GetEmailIdentityPolicies
	ses: GetEmailTemplate
	ses: GetIdentityDkimAttributes
	ses: GetIdentityMailFromDomainAttributes
	ses: GetIdentityNotificationAttributes
	ses: GetIdentityPolicies
	ses: GetIdentityVerificationAttributes
	ses: GetImportJob
	ses: GetIngressPoint
	ses: GetMemberOfAddressList
	ses: GetMessageInsights
	ses: GetMultiRegionEndpoint
	ses: GetRelay
	ses: GetRuleSet
	ses: GetSendQuota

Awalan layanan	Tindakan
	ses: GetSendStatistics
	ses: GetSuppressedDestination
	ses: GetTemplate
	ses: GetTenant
	ses: GetTrafficPolicy
	ses: ListAddonInstances
	ses: ListAddonSubscriptions
	ses: ListAddressListImportJobs
	ses: ListAddressLists
	ses: ListArchiveExports
	ses: ListArchiveSearches
	ses: ListArchives
	ses: ListConfigurationSets
	ses: ListContactLists
	ses: ListContacts
	ses: ListCustomVerificationEmailTemplates
	ses: ListDedicatedIpPools
	ses: ListDeliverabilityTestReports
	ses: ListDomainDeliverabilityCampaigns
	ses: ListEmailIdentities
	ses: ListEmailTemplates

Awalan layanan	Tindakan
	ses: ListExportJobs
	ses: ListIdentities
	ses: ListIdentityPolicies
	ses: ListImportJobs
	ses: ListIngressPoints
	ses: ListMembersOfAddressList
	ses: ListMultiRegionEndpoints
	ses: ListReceiptFilters
	ses: ListReceiptRuleSets
	ses: ListRecommendations
	ses: ListRelays
	ses: ListReputationEntities
	ses: ListResourceTenants
	ses: ListRuleSets
	ses: ListSuppressedDestinations
	ses: ListTemplates
	ses: ListTenantResources
	ses: ListTenants
	ses: ListTrafficPolicies
	ses: ListVerifiedEmailAddresses
	ses: PutAccountDedicatedIpWarmupAttributes

Awalan layanan	Tindakan
	ses: PutAccountDetails
	ses: PutAccountSendingAttributes
	ses: PutAccountSuppressionAttributes
	ses: PutAccountVdmAttributes
	ses: PutConfigurationSetArchivingOptions
	ses: PutConfigurationSetDeliveryOptions
	ses: PutConfigurationSetReputationOptions
	ses: PutConfigurationSetSendingOptions
	ses: PutConfigurationSetSuppressionOptions
	ses: PutConfigurationSetTrackingOptions
	ses: PutConfigurationSetVdmOptions
	ses: PutDedicatedIpInPool
	ses: PutDedicatedIpPoolScalingAttributes
	ses: PutDedicatedIpWarmupAttributes
	ses: PutDeliverabilityDashboardOption
	ses: PutEmailIdentityConfigurationSetAttributes
	ses: PutEmailIdentityDkimAttributes
	ses: PutEmailIdentityDkimSigningAttributes
	ses: PutEmailIdentityFeedbackAttributes
	ses: PutEmailIdentityMailFromAttributes
	ses: PutIdentityPolicy

Awalan layanan	Tindakan
	ses: PutSuppressedDestination
	ses: PutTenantSuppressionAttributes
	ses: RegisterMemberToAddressList
	ses: ReorderReceiptRuleSet
	ses: SendBounce
	ses: SendCustomVerificationEmail
	ses: SetActiveReceiptRuleSet
	ses: SetIdentityDkimEnabled
	ses: SetIdentityFeedbackForwardingEnabled
	ses: SetIdentityHeadersInNotificationsEnabled
	ses: SetIdentityMailFromDomain
	ses: SetIdentityNotificationTopic
	ses: SetReceiptRulePosition
	ses: StartArchiveExport
	ses: StartArchiveSearch
	ses: StopArchiveExport
	ses: StopArchiveSearch
	ses: TestRenderEmailTemplate
	ses: TestRenderTemplate
	ses: UpdateAccountSendingEnabled
	ses: UpdateArchive

Awalan layanan	Tindakan
	ses: UpdateConfigurationSetEventDestination
	ses: UpdateConfigurationSetReputationMetricsEnabled
	ses: UpdateConfigurationSetSendingEnabled
	ses: UpdateConfigurationSetTrackingOptions
	ses: UpdateContact
	ses: UpdateContactList
	ses: UpdateCustomVerificationEmailTemplate
	ses: UpdateEmailIdentityPolicy
	ses: UpdateEmailTemplate
	ses: UpdateIngressPoint
	ses: UpdateReceiptRule
	ses: UpdateRelay
	ses: UpdateRuleSet
	ses: UpdateTemplate
	ses: UpdateTrafficPolicy
	ses: VerifyDomainDkim
	ses: VerifyDomainIdentity
	ses: VerifyEmailAddress
	ses: VerifyEmailIdentity

Awalan layanan	Tindakan
melindungi	perisai: AssociateDRTLogBucket perisai: AssociateHealthCheck perisai: AssociateProactiveEngagementDetails perisai: CreateProtection perisai: CreateProtectionGroup perisai: CreateSubscription perisai: DeleteProtection perisai: DeleteProtectionGroup perisai: DeleteSubscription perisai: DescribeAttack perisai: DescribeAttackStatistics Shield:DescribedrtAccess perisai: DescribeEmergencyContactSettings perisai: DescribeProtection perisai: DescribeProtectionGroup perisai: DescribeSubscription perisai: DisableApplicationLayerAutomaticResponse perisai: DisableProactiveEngagement perisai: DisassociateDRTLogBucket Perisai: DisassociateDrTrole perisai: DisassociateHealthCheck

Awalan layanan	Tindakan
	perisai: EnableApplicationLayerAutomaticResponse perisai: EnableProactiveEngagement perisai: GetSubscriptionState perisai: ListAttacks perisai: ListProtectionGroups perisai: ListProtections perisai: ListResourcesInProtectionGroup perisai: UpdateApplicationLayerAutomaticResponse perisai: UpdateEmergencyContactSettings perisai: UpdateProtectionGroup perisai: UpdateSubscription

Awalan layanan	Tindakan
penandatangan	penandatangan: AddProfilePermission
	penandatangan: CancelSigningProfile
	penandatangan: DescribeSigningJob
	penandatangan: GetSigningPlatform
	penandatangan: GetSigningProfile
	penandatangan: ListProfilePermissions
	penandatangan: ListSigningJobs
	penandatangan: ListSigningPlatforms
	penandatangan: ListSigningProfiles
	penandatangan: PutSigningProfile
	penandatangan: RemoveProfilePermission
	penandatangan: RevokeSignature
	penandatangan: RevokeSigningProfile
	penandatangan: SignPayload
	penandatangan: StartSigningJob

Awalan layanan	Tindakan
simspaceweaver	simspaceweaver: CreateSnapshot simspaceweaver: DeleteApp simspaceweaver: DeleteSimulation simspaceweaver: DescribeApp simspaceweaver: DescribeSimulation simspaceweaver: ListApps simspaceweaver: ListSimulations simspaceweaver: StartApp simspaceweaver: StartClock simspaceweaver: StartSimulation simspaceweaver: StopApp simspaceweaver: StopClock simspaceweaver: StopSimulation

Awalan layanan	Tindakan
sms	sms: CreateApp
	sms: CreateReplicationJob
	sms: DeleteApp
	sms: DeleteAppLaunchConfiguration
	sms: DeleteAppReplicationConfiguration
	sms: DeleteAppValidationConfiguration
	sms: DeleteReplicationJob
	sms: DeleteServerCatalog
	sms: DisassociateConnector
	sms: GenerateChangeSet
	sms: GenerateTemplate
	sms: GetApp
	sms: GetAppLaunchConfiguration
	sms: GetAppReplicationConfiguration
	sms: GetAppValidationConfiguration
	sms: GetAppValidationOutput
	sms: GetConnectors
	sms: GetReplicationJobs
	sms: GetReplicationRuns
	sms: GetServers
	sms: ImportAppCatalog

Awalan layanan	Tindakan
	sms: ImportServerCatalog
	sms: LaunchApp
	sms: ListApps
	sms: NotifyAppValidationOutput
	sms: PutAppLaunchConfiguration
	sms: PutAppReplicationConfiguration
	sms: PutAppValidationConfiguration
	sms: StartAppReplication
	sms: StartOnDemandAppReplication
	sms: StartOnDemandReplicationRun
	sms: StopAppReplication
	sms: TerminateApp
	sms: UpdateApp
	sms: UpdateReplicationJob

Awalan layanan	Tindakan
sms-suara	sms-suara: AssociateProtectConfiguration sms-suara: CreateConfigurationSet sms-suara: CreateConfigurationSetEventDestination sms-suara: CreateEventDestination sms-suara: CreateOptOutList sms-suara: CreatePool sms-suara: CreateProtectConfiguration sms-suara: CreateRegistration sms-suara: CreateRegistrationAssociation sms-suara: CreateRegistrationAttachment sms-suara: CreateRegistrationVersion sms-suara: CreateVerifiedDestinationNumber sms-suara: DeleteAccountDefaultProtectConfiguration sms-suara: DeleteConfigurationSet sms-suara: DeleteConfigurationSetEventDestination sms-suara: DeleteDefaultMessageType sms-suara: DeleteDefaultSenderId sms-suara: DeleteEventDestination sms-suara: DeleteKeyword sms-suara: DeleteMediaMessageSpendLimitOverride sms-suara: DeleteOptOutList

Awalan layanan	Tindakan
	sms-suara: DeleteOptedOutNumber sms-suara: DeletePool sms-suara: DeleteProtectConfiguration sms-suara: DeleteProtectConfigurationRuleSetNumberOverride sms-suara: DeleteRegistration sms-suara: DeleteRegistrationAttachment sms-suara: DeleteResourcePolicy sms-suara: DeleteTextMessageSpendLimitOverride sms-suara: DeleteVerifiedDestinationNumber sms-suara: DeleteVoiceMessageSpendLimitOverride sms-suara: DescribeAccountAttributes sms-suara: DescribeAccountLimits sms-suara: DescribeConfigurationSets sms-suara: DescribeKeywords sms-suara: DescribeOptOutLists sms-suara: DescribeOptedOutNumbers sms-suara: DescribePhoneNumbers sms-suara: DescribePools sms-suara: DescribeProtectConfigurations sms-suara: DescribeRegistrationAttachments sms-suara: DescribeRegistrationFieldDefinitions

Awalan layanan	Tindakan
	sms-suara: DescribeRegistrationFieldValues sms-suara: DescribeRegistrationSectionDefinitions sms-suara: DescribeRegistrationTypeDefinitions sms-suara: DescribeRegistrationVersions sms-suara: DescribeRegistrations sms-suara: DescribeSenderIds sms-suara: DescribeSpendLimits sms-suara: DescribeVerifiedDestinationNumbers sms-suara: DisassociateOriginationIdentity sms-suara: DisassociateProtectConfiguration sms-suara: DiscardRegistrationVersion sms-suara: GetConfigurationSetEventDestinations sms-suara: GetProtectConfigurationCountryRuleSet sms-suara: GetResourcePolicy sms-suara: ListConfigurationSets sms-suara: ListPoolOriginationIdentities sms-suara: ListProtectConfigurationRuleSetNumberOverrides sms-suara: ListRegistrationAssociations sms-suara: PutKeyword sms-suara: PutOptedOutNumber sms-suara: PutProtectConfigurationRuleSetNumberOverride

Awalan layanan	Tindakan
	sms-suara: PutResourcePolicy sms-suara: ReleasePhoneNumber sms-suara: ReleaseSenderId sms-suara: RequestPhoneNumber sms-suara: RequestSenderId sms-suara: SendDestinationNumberVerificationCode sms-suara: SetAccountDefaultProtectConfiguration sms-suara: SetDefaultMessageFeedbackEnabled sms-suara: SetDefaultMessageType sms-suara: SetDefaultSenderId sms-suara: SetMediaMessageSpendLimitOverride sms-suara: SetTextMessageSpendLimitOverride sms-suara: SetVoiceMessageSpendLimitOverride sms-suara: SubmitRegistrationVersion sms-suara: UpdateConfigurationSetEventDestination sms-suara: UpdateEventDestination sms-suara: UpdatePhoneNumber sms-suara: UpdatePool sms-suara: UpdateProtectConfiguration sms-suara: UpdateProtectConfigurationCountryRuleSet sms-suara: UpdateSenderId

Awalan layanan	Tindakan
bola salju	bola salju: CancelCluster
	bola salju: CancelJob
	bola salju: CreateAddress
	bola salju: CreateCluster
	bola salju: CreateJob
	bola salju: CreateLongTermPricing
	bola salju: CreateReturnShippingLabel
	bola salju: DescribeAddress
	bola salju: DescribeAddresses
	bola salju: DescribeCluster
	bola salju: DescribeJob
	bola salju: DescribeReturnShippingLabel
	bola salju: GetJobManifest
	bola salju: GetJobUnlockCode
	bola salju: GetSnowballUsage
	bola salju: GetSoftwareUpdates
	bola salju: ListClusterJobs
	bola salju: ListClusters
	bola salju: ListCompatibleImages
	bola salju: ListJobs
	bola salju: ListLongTermPricing

Awalan layanan	Tindakan
	bola salju: ListPickupLocations bola salju: ListServiceVersions bola salju: UpdateCluster bola salju: UpdateJob bola salju: UpdateJobShipmentState bola salju: UpdateLongTermPricing
sqs	persegi: AddPermission persegi: CancelMessageMoveTask persegi: CreateQueue persegi: DeleteQueue persegi: PurgeQueue persegi: RemovePermission persegi: SetQueueAttributes

Awalan layanan	Tindakan
ssm	ssm: AssociateOpsItemRelatedItem
	ssm: CancelCommand
	ssm: CancelMaintenanceWindowExecution
	ssm: CreateActivation
	ssm: CreateAssociation
	ssm: CreateAssociationBatch
	ssm: CreateDocument
	ssm: CreateMaintenanceWindow
	ssm: CreateOpsItem
	ssm: CreateOpsMetadata
	ssm: CreatePatchBaseline
	ssm: CreateResourceDataSync
	ssm: DeleteActivation
	ssm: DeleteAssociation
	ssm: DeleteCloudConnector
	ssm: DeleteDocument
	ssm: DeleteInventory
	ssm: DeleteMaintenanceWindow
	ssm: DeleteOpsItem
	ssm: DeleteOpsMetadata
	ssm: DeleteParameter

Awalan layanan	Tindakan
	ssm: DeleteParameters
	ssm: DeletePatchBaseline
	ssm: DeleteResourceDataSync
	ssm: DeleteResourcePolicy
	ssm: DeregisterManagedInstance
	ssm: DeregisterPatchBaselineForPatchGroup
	ssm: DeregisterTargetFromMaintenanceWindow
	ssm: DeregisterTaskFromMaintenanceWindow
	ssm: DescribeActivations
	ssm: DescribeAssociation
	ssm: DescribeAssociationExecutionTargets
	ssm: DescribeAssociationExecutions
	ssm: DescribeAutomationExecutions
	ssm: DescribeAutomationStepExecutions
	ssm: DescribeAvailablePatches
	ssm: DescribeDocument
	ssm: DescribeDocumentParameters
	ssm: DescribeDocumentPermission
	ssm: DescribeEffectiveInstanceAssociations
	ssm: DescribeEffectivePatchesForPatchBaseline
	ssm: DescribeInstanceAssociationsStatus

Awalan layanan	Tindakan
	ssm: DescribeInstanceInformation
	ssm: DescribeInstancePatchStates
	ssm: DescribeInstancePatchStatesForPatchGroup
	ssm: DescribeInstancePatches
	ssm: DescribeInstanceProperties
	ssm: DescribeInventoryDeletions
	ssm: DescribeMaintenanceWindowExecutionTaskInvocations
	ssm: DescribeMaintenanceWindowExecutionTasks
	ssm: DescribeMaintenanceWindowExecutions
	ssm: DescribeMaintenanceWindowSchedule
	ssm: DescribeMaintenanceWindowTargets
	ssm: DescribeMaintenanceWindowTasks
	ssm: DescribeMaintenanceWindows
	ssm: DescribeMaintenanceWindowsForTarget
	ssm: DescribeOpsItems
	ssm: DescribeParameters
	ssm: DescribePatchBaselines
	ssm: DescribePatchGroupState
	ssm: DescribePatchGroups
	ssm: DescribePatchProperties
	ssm: DescribeSessions

Awalan layanan	Tindakan
	ssm: DisassociateOpsItemRelatedItem
	ssm: GetAccessToken
	ssm: GetAutomationExecution
	ssm: GetCalendarState
	ssm: GetCloudConnector
	ssm: GetCommandInvocation
	ssm: GetConnectionStatus
	ssm: GetDefaultPatchBaseline
	ssm: GetDeployablePatchSnapshotForInstance
	ssm: GetDocument
	ssm: GetExecutionPreview
	ssm: GetInventory
	ssm: GetInventorySchema
	ssm: GetMaintenanceWindow
	ssm: GetMaintenanceWindowExecution
	ssm: GetMaintenanceWindowExecutionTask
	ssm: GetMaintenanceWindowExecutionTaskInvocation
	ssm: GetMaintenanceWindowTask
	ssm: GetOpsItem
	ssm: GetOpsMetadata
	ssm: GetOpsSummary

Awalan layanan	Tindakan
	ssm: GetParameter
	ssm: GetParameterHistory
	ssm: GetParameters
	ssm: GetParametersByPath
	ssm: GetPatchBaseline
	ssm: GetPatchBaselineForPatchGroup
	ssm: GetResourcePolicies
	ssm: GetServiceSetting
	ssm: LabelParameterVersion
	ssm: ListAssociationVersions
	ssm: ListAssociations
	ssm: ListCloudConnectors
	ssm: ListCommandInvocations
	ssm: ListCommands
	ssm: ListComplianceItems
	ssm: ListComplianceSummaries
	ssm: ListDocumentMetadataHistory
	ssm: ListDocumentVersions
	ssm: ListDocuments
	ssm: ListInstanceAssociations
	ssm: ListInventoryEntries

Awalan layanan	Tindakan
	ssm: ListNodes
	ssm: ListNodesSummary
	ssm: ListOpsItemEvents
	ssm: ListOpsItemRelatedItems
	ssm: ListOpsMetadata
	ssm: ListResourceComplianceSummaries
	ssm: ListResourceDataSync
	ssm: ModifyDocumentPermission
	ssm: PutComplianceItems
	ssm: PutInventory
	ssm: PutParameter
	ssm: PutResourcePolicy
	ssm: RegisterDefaultPatchBaseline
	ssm: RegisterManagedInstance
	ssm: RegisterPatchBaselineForPatchGroup
	ssm: RegisterTargetWithMaintenanceWindow
	ssm: RegisterTaskWithMaintenanceWindow
	ssm: ResetServiceSetting
	ssm: ResumeSession
	ssm: SendAutomationSignal
	ssm: SendCommand

Awalan layanan	Tindakan
	ssm: StartAssociationsOnce
	ssm: StartAutomationExecution
	ssm: StartChangeRequestExecution
	ssm: StartSession
	ssm: StopAutomationExecution
	ssm: TerminateSession
	ssm: UnlabelParameterVersion
	ssm: UpdateAssociation
	ssm: UpdateAssociationStatus
	ssm: UpdateCloudConnector
	ssm: UpdateDocument
	ssm: UpdateDocumentDefaultVersion
	ssm: UpdateDocumentMetadata
	ssm: UpdateInstanceInformation
	ssm: UpdateMaintenanceWindow
	ssm: UpdateMaintenanceWindowTarget
	ssm: UpdateMaintenanceWindowTask
	ssm: UpdateManagedInstanceRole
	ssm: UpdateOpsItem
	ssm: UpdateOpsMetadata
	ssm: UpdatePatchBaseline

Awalan layanan	Tindakan
	ssm: UpdateResourceDataSync ssm: UpdateServiceSetting ssm: ValidateCloudConnector

Awalan layanan	Tindakan
insiden ssm	insiden ssm-: BatchGetIncidentFindings insiden ssm-: CreateReplicationSet insiden ssm-: CreateResponsePlan insiden ssm-: CreateTimelineEvent insiden ssm-: DeleteIncidentRecord insiden ssm-: DeleteReplicationSet insiden ssm-: DeleteResourcePolicy insiden ssm-: DeleteResponsePlan insiden ssm-: DeleteTimelineEvent insiden ssm-: GetIncidentRecord insiden ssm-: GetReplicationSet insiden ssm-: GetResourcePolicies insiden ssm-: GetResponsePlan insiden ssm-: GetTimelineEvent insiden ssm-: ListIncidentFindings insiden ssm-: ListIncidentRecords insiden ssm-: ListRelatedItems insiden ssm-: ListReplicationSets insiden ssm-: ListResponsePlans insiden ssm-: ListTimelineEvents insiden ssm-: PutResourcePolicy

Awalan layanan	Tindakan
	insiden ssm-: StartIncident insiden ssm-: UpdateDeletionProtection insiden ssm-: UpdateIncidentRecord insiden ssm-: UpdateRelatedItems insiden ssm-: UpdateReplicationSet insiden ssm-: UpdateResponsePlan insiden ssm-: UpdateTimelineEvent

Awalan layanan	Tindakan
ssm-sap	ssm-sap: BackupDatabase
	ssm-sap: DeleteResourcePermission
	ssm-sap: DeregisterApplication
	ssm-sap: GetApplication
	ssm-sap: GetComponent
	ssm-sap: GetConfigurationCheckOperation
	ssm-sap: GetDatabase
	ssm-sap: GetOperation
	ssm-sap: GetResourcePermission
	ssm-sap: ListApplications
	ssm-sap: ListComponents
	ssm-sap: ListConfigurationCheckDefinitions
	ssm-sap: ListConfigurationCheckOperations
	ssm-sap: ListDatabases
	ssm-sap: ListOperationEvents
	ssm-sap: ListOperations
	ssm-sap: ListSubCheckResults
	ssm-sap: ListSubCheckRuleResults
	ssm-sap: PutResourcePermission
	ssm-sap: RegisterApplication
	ssm-sap: RestoreDatabase

Awalan layanan	Tindakan
	ssm-sap: StartApplication
	ssm-sap: StartApplicationRefresh
	ssm-sap: StartConfigurationChecks
	ssm-sap: StopApplication
	ssm-sap: UpdateApplicationSettings
	ssm-sap: UpdateHANABackupSettings

Awalan layanan	Tindakan
negara	negara: CreateActivity negara: CreateStateMachine negara: CreateStateMachineAlias negara: DeleteActivity negara: DeleteStateMachine negara: DeleteStateMachineAlias negara: DeleteStateMachineVersion negara: DescribeActivity negara: DescribeExecution negara: DescribeMapRun negara: DescribeStateMachine negara: DescribeStateMachineAlias negara: DescribeStateMachineForExecution negara: GetExecutionHistory negara: ListActivities negara: ListExecutions negara: ListMapRuns negara: ListStateMachineAliases negara: ListStateMachineVersions negara: ListStateMachines negara: SendTaskFailure

Awalan layanan	Tindakan
	negara bagian: SendTaskHeartbeat
	negara bagian: SendTaskSuccess
	negara bagian: StartExecution
	negara bagian: StopExecution
	negara bagian: UpdateMapRun
	negara bagian: UpdateStateMachine
	negara bagian: UpdateStateMachineAlias
	negara bagian: ValidateStateMachineDefinition
sts	sts: AssumeRole
	sts: AssumeRoleWithSAML
	sts: AssumeRoleWithWebIdentity
	sts: DecodeAuthorizationMessage
	sts: GetAccessKeyInfo
	sts: GetCallerIdentity
	sts: GetFederationToken
	sts: GetSessionToken
	sts: GetWebIdentityToken

Awalan layanan	Tindakan
swf	swf: DeleteActivityType swf: DeleteWorkflowType swf: DeprecateActivityType swf: DeprecateDomain swf: DeprecateWorkflowType swf: DescribeActivityType swf: DescribeDomain swf: DescribeWorkflowType swf: ListActivityTypes swf: ListDomains swf: ListWorkflowTypes swf: RegisterActivityType swf: RegisterDomain swf: RegisterWorkflowType swf: UndeprecateActivityType swf: UndeprecateDomain swf: UndeprecateWorkflowType

Awalan layanan	Tindakan
sintetis	sintetis: AssociateResource
	sintetis: CreateCanary
	sintetis: CreateGroup
	sintetis: DeleteCanary
	sintetis: DeleteGroup
	sintetis: DescribeCanaries
	sintetis: DescribeCanariesLastRun
	sintetis: DescribeRuntimeVersions
	sintetis: DisassociateResource
	sintetis: GetCanary
	sintetis: GetCanaryRuns
	sintetis: GetGroup
	sintetis: ListAssociatedGroups
	sintetis: ListGroupResources
	sintetis: ListGroups
	sintetis: StartCanary
	sintetis: StartCanaryDryRun
	sintetis: StopCanary
	sintetis: UpdateCanary

Awalan layanan	Tindakan
tanda	Tag: DescribeReportCreation Tag: GetComplianceSummary Tag: GetResources Tag: StartReportCreation

Awalan layanan	Tindakan
textract	teks: AnalyzeDocument
	teks: AnalyzeExpense
	Textract:AnalyzeID
	teks: CreateAdapter
	teks: CreateAdapterVersion
	teks: DeleteAdapter
	teks: DeleteAdapterVersion
	teks: DetectDocumentText
	teks: GetAdapter
	teks: GetAdapterVersion
	teks: GetDocumentAnalysis
	teks: GetDocumentTextDetection
	teks: GetExpenseAnalysis
	teks: GetLendingAnalysis
	teks: GetLendingAnalysisSummary
	teks: ListAdapterVersions
	teks: ListAdapters
	teks: StartDocumentAnalysis
	teks: StartDocumentTextDetection
	teks: StartExpenseAnalysis
	teks: StartLendingAnalysis

Awalan layanan	Tindakan
	teks: UpdateAdapter

Awalan layanan	Tindakan
aliran waktu	aliran waktu: CancelQuery aliran waktu: CreateDatabase aliran waktu: CreateScheduledQuery aliran waktu: CreateTable aliran waktu: DeleteDatabase aliran waktu: DeleteScheduledQuery aliran waktu: DeleteTable aliran waktu: DescribeAccountSettings aliran waktu: DescribeDatabase aliran waktu: DescribeScheduledQuery aliran waktu: DescribeTable aliran waktu: ExecuteScheduledQuery aliran waktu: ListBatchLoadTasks aliran waktu: ListDatabases aliran waktu: ListScheduledQueries aliran waktu: ListTables aliran waktu: PrepareQuery aliran waktu: UpdateAccountSettings aliran waktu: UpdateDatabase aliran waktu: UpdateScheduledQuery aliran waktu: UpdateTable

Awalan layanan	Tindakan
tnb	tnb: CancelSolNetworkOperation
	tnb: CreateSolFunctionPackage
	tnb: CreateSolNetworkInstance
	tnb: CreateSolNetworkPackage
	tnb: DeleteSolFunctionPackage
	tnb: DeleteSolNetworkInstance
	tnb: DeleteSolNetworkPackage
	tnb: GetSolFunctionInstance
	tnb: GetSolFunctionPackage
	tnb: GetSolFunctionPackageContent
	tnb: GetSolFunctionPackageDescriptor
	tnb: GetSolNetworkInstance
	tnb: GetSolNetworkOperation
	tnb: GetSolNetworkPackage
	tnb: GetSolNetworkPackageContent
	tnb: GetSolNetworkPackageDescriptor
	tnb: InstantiateSolNetworkInstance
	tnb: ListSolFunctionInstances
	tnb: ListSolFunctionPackages
	tnb: ListSolNetworkInstances
	tnb: ListSolNetworkOperations

Awalan layanan	Tindakan
	tnb: ListSolNetworkPackages
	tnb: PutSolFunctionPackageContent
	tnb: PutSolNetworkPackageContent
	tnb: TerminateSolNetworkInstance
	tnb: UpdateSolFunctionPackage
	tnb: UpdateSolNetworkInstance
	tnb: UpdateSolNetworkPackage
	tnb: ValidateSolFunctionPackageContent
	tnb: ValidateSolNetworkPackageContent

Awalan layanan	Tindakan
mentranskripsikan	transkripsikan: CreateCallAnalyticsCategory transkripsikan: CreateLanguageModel transkripsikan: CreateMedicalVocabulary transkripsikan: CreateVocabulary transkripsikan: CreateVocabularyFilter transkripsikan: DeleteCallAnalyticsCategory transkripsikan: DeleteCallAnalyticsJob transkripsikan: DeleteLanguageModel transkripsikan: DeleteMedicalScribeJob transkripsikan: DeleteMedicalTranscriptionJob transkripsikan: DeleteMedicalVocabulary transkripsikan: DeleteTranscriptionJob transkripsikan: DeleteVocabulary transkripsikan: DeleteVocabularyFilter transkripsikan: DescribeLanguageModel transkripsikan: GetCallAnalyticsCategory transkripsikan: GetCallAnalyticsJob transkripsikan: GetMedicalScribeJob transkripsikan: GetMedicalTranscriptionJob transkripsikan: GetMedicalVocabulary transkripsikan: GetTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: GetVocabulary transkripsikan: GetVocabularyFilter transkripsikan: ListCallAnalyticsCategories transkripsikan: ListCallAnalyticsJobs transkripsikan: ListLanguageModels transkripsikan: ListMedicalScribeJobs transkripsikan: ListMedicalTranscriptionJobs transkripsikan: ListMedicalVocabularies transkripsikan: ListTranscriptionJobs transkripsikan: ListVocabularies transkripsikan: ListVocabularyFilters transkripsikan: StartCallAnalyticsJob transkripsikan: StartCallAnalyticsStreamTranscription transkripsikan: StartCallAnalyticsStreamTranscriptionWebSocket transkripsikan: StartMedicalScribeJob transkripsikan: StartMedicalStreamTranscription transkripsikan: StartMedicalStreamTranscriptionWebSocket transkripsikan: StartMedicalTranscriptionJob transkripsikan: StartStreamTranscription transkripsikan: StartStreamTranscriptionWebSocket transkripsikan: StartTranscriptionJob

Awalan layanan	Tindakan
	transkripsikan: UpdateCallAnalyticsCategory transkripsikan: UpdateMedicalVocabulary transkripsikan: UpdateVocabulary transkripsikan: UpdateVocabularyFilter

Awalan layanan	Tindakan
transfer	transfer: CreateAccess
	transfer: CreateAgreement
	transfer: CreateConnector
	transfer: CreateProfile
	transfer: CreateServer
	transfer: CreateUser
	transfer: CreateWebApp
	transfer: CreateWorkflow
	transfer: DeleteAccess
	transfer: DeleteAgreement
	transfer: DeleteCertificate
	transfer: DeleteConnector
	transfer: DeleteHostKey
	transfer: DeleteProfile
	transfer: DeleteServer
	transfer: DeleteSshPublicKey
	transfer: DeleteUser
	transfer: DeleteWebApp
	transfer: DeleteWebAppCustomization
	transfer: DeleteWorkflow
	transfer: DescribeAccess

Awalan layanan	Tindakan
	transfer: DescribeAgreement
	transfer: DescribeCertificate
	transfer: DescribeConnector
	transfer: DescribeExecution
	transfer: DescribeHostKey
	transfer: DescribeProfile
	transfer: DescribeSecurityPolicy
	transfer: DescribeServer
	transfer: DescribeUser
	transfer: DescribeWebApp
	transfer: DescribeWebAppCustomization
	transfer: DescribeWorkflow
	transfer: ImportCertificate
	transfer: ImportHostKey
	transfer: ImportSshPublicKey
	transfer: ListAccesses
	transfer: ListCertificates
	transfer: ListConnectors
	transfer: ListExecutions
	transfer: ListFileTransferResults
	transfer: ListHostKeys

Awalan layanan	Tindakan
	transfer: ListProfiles
	transfer: ListSecurityPolicies
	transfer: ListServers
	transfer: ListUsers
	transfer: ListWebApps
	transfer: ListWorkflows
	transfer: SendWorkflowStepState
	transfer: StartDirectoryListing
	transfer: StartFileTransfer
	transfer: StartRemoteDelete
	transfer: StartRemoteMove
	transfer: StartServer
	transfer: StopServer
	transfer: TestConnection
	transfer: TestIdentityProvider
	transfer: UpdateAccess
	transfer: UpdateAgreement
	transfer: UpdateCertificate
	transfer: UpdateConnector
	transfer: UpdateHostKey
	transfer: UpdateProfile

Awalan layanan	Tindakan
	transfer: UpdateServer transfer: UpdateUser transfer: UpdateWebApp transfer: UpdateWebAppCustomization
menerjemahkan	menerjemahkan: CreateParallelData menerjemahkan: DeleteParallelData menerjemahkan: DeleteTerminology menerjemahkan: DescribeTextTranslationJob menerjemahkan: GetParallelData menerjemahkan: GetTerminology menerjemahkan: ImportTerminology menerjemahkan: ListLanguages menerjemahkan: ListParallelData menerjemahkan: ListTerminologies menerjemahkan: ListTextTranslationJobs menerjemahkan: StartTextTranslationJob menerjemahkan: StopTextTranslationJob menerjemahkan: TranslateDocument menerjemahkan: TranslateText menerjemahkan: UpdateParallelData

Awalan layanan	Tindakan
voiceid	voiceid: AssociateFraudster
	voiceid: CreateDomain
	voiceid: CreateWatchlist
	voiceid: DeleteDomain
	voiceid: DeleteFraudster
	voiceid: DeleteSpeaker
	voiceid: DeleteWatchlist
	voiceid: DescribeDomain
	voiceid: DescribeFraudster
	voiceid: DescribeFraudsterRegistrationJob
	voiceid: DescribeSpeaker
	voiceid: DescribeSpeakerEnrollmentJob
	voiceid: DescribeWatchlist
	voiceid: DisassociateFraudster
	voiceid: EvaluateSession
	voiceid: ListDomains
	voiceid: ListFraudsterRegistrationJobs
	voiceid: ListFraudsters
	voiceid: ListSpeakerEnrollmentJobs
	voiceid: ListSpeakers
	voiceid: ListWatchlists

Awalan layanan	Tindakan
	voiceid: OptOutSpeaker voiceid: StartFraudsterRegistrationJob voiceid: StartSpeakerEnrollmentJob voiceid: UpdateDomain voiceid: UpdateWatchlist

Awalan layanan	Tindakan
vpc-kisi	vpc-kisi: CreateAccessLogSubscription vpc-kisi: CreateListener vpc-kisi: CreateResourceConfiguration vpc-kisi: CreateResourceGateway vpc-kisi: CreateRule vpc-kisi: CreateService vpc-kisi: CreateServiceNetwork vpc-kisi: CreateServiceNetworkResourceAssociation vpc-kisi: CreateServiceNetworkServiceAssociation vpc-kisi: CreateServiceNetworkVpcAssociation vpc-kisi: CreateTargetGroup vpc-kisi: DeleteAccessLogSubscription vpc-kisi: DeleteAuthPolicy vpc-kisi: DeleteDomainVerification vpc-kisi: DeleteListener vpc-kisi: DeleteResourceConfiguration vpc-kisi: DeleteResourceEndpointAssociation vpc-kisi: DeleteResourceGateway vpc-kisi: DeleteResourcePolicy vpc-kisi: DeleteRule vpc-kisi: DeleteService

Awalan layanan	Tindakan
	vpc-kisi: DeleteServiceNetwork vpc-kisi: DeleteServiceNetworkResourceAssociation vpc-kisi: DeleteServiceNetworkServiceAssociation vpc-kisi: DeleteServiceNetworkVpcAssociation vpc-kisi: DeleteTargetGroup vpc-kisi: DeregisterTargets vpc-kisi: GetAccessLogSubscription vpc-kisi: GetAuthPolicy vpc-kisi: GetDomainVerification vpc-kisi: GetListener vpc-kisi: GetResourceConfiguration vpc-kisi: GetResourceGateway vpc-kisi: GetResourcePolicy vpc-kisi: GetRule vpc-kisi: GetService vpc-kisi: GetServiceNetwork vpc-kisi: GetServiceNetworkResourceAssociation vpc-kisi: GetServiceNetworkServiceAssociation vpc-kisi: GetServiceNetworkVpcAssociation vpc-kisi: GetTargetGroup vpc-kisi: ListAccessLogSubscriptions

Awalan layanan	Tindakan
	vpc-kisi: ListDomainVerifications vpc-kisi: ListListeners vpc-kisi: ListResourceConfigurations vpc-kisi: ListResourceEndpointAssociations vpc-kisi: ListResourceGateways vpc-kisi: ListRules vpc-kisi: ListServiceNetworkResourceAssociations vpc-kisi: ListServiceNetworkServiceAssociations vpc-kisi: ListServiceNetworkVpcAssociations vpc-kisi: ListServiceNetworkVpcEndpointAssociations vpc-kisi: ListServiceNetworks vpc-kisi: ListServices vpc-kisi: ListTargetGroups vpc-kisi: ListTargets vpc-kisi: PutAuthPolicy vpc-kisi: PutResourcePolicy vpc-kisi: RegisterTargets vpc-kisi: StartDomainVerification vpc-kisi: UpdateAccessLogSubscription vpc-kisi: UpdateListener vpc-kisi: UpdateResourceConfiguration

Awalan layanan	Tindakan
	vpc-kisi: UpdateResourceGateway vpc-kisi: UpdateRule vpc-kisi: UpdateService vpc-kisi: UpdateServiceNetwork vpc-kisi: UpdateServiceNetworkVpcAssociation vpc-kisi: UpdateTargetGroup

Awalan layanan	Tindakan
wafv2	wafv2: ACL AssociateWeb
	wafv2: CheckCapacity
	WAFv2:createApiKey
	WAFv2: CreateIPSet
	wafv2: CreateRegexPatternSet
	wafv2: CreateRuleGroup
	wafv2: ACL CreateWeb
	WAFv2: DeleteApiKey
	wafv2: DeleteFirewallManagerRuleGroups
	WAFv2: DeleteIPSet
	wafv2: DeleteLoggingConfiguration
	wafv2: DeletePermissionPolicy
	wafv2: DeleteRegexPatternSet
	wafv2: DeleteRuleGroup
	wafv2: ACL DeleteWeb
	wafv2: DescribeAllManagedProducts
	wafv2: DescribeManagedProductsByVendor
	wafv2: DescribeManagedRuleGroup
	wafv2: ACL DisassociateWeb
	wafv2: GenerateMobileSdkReleaseUrl
	wafv2: GetDecryptedAPIKey

Awalan layanan	Tindakan
	WAFV2: GetIpSet
	wafv2: GetLoggingConfiguration
	wafv2: GetManagedRuleSet
	wafv2: GetMobileSdkRelease
	wafv2: GetRateBasedStatementManagedKeys
	wafv2: GetRegexPatternSet
	wafv2: GetRevenueStatistics
	wafv2: GetRevenueStatisticsSummary
	wafv2: GetRevenueStatisticsTimeSeries
	wafv2: GetRuleGroup
	wafv2: GetSampledRequests
	wafv2: GetWebACLForResource
	WAFv2:ListApiKeys
	wafv2: ListAvailableManagedRuleGroupVersions
	wafv2: ListAvailableManagedRuleGroups
	WAFv2: ListipSets
	wafv2: ListLoggingConfigurations
	wafv2: ListManagedRuleSets
	wafv2: ListMobileSdkReleases
	wafv2: ListRegexPatternSets
	wafv2: ListResourcesForWebACL

Awalan layanan	Tindakan
	wafv2: ListRuleGroups
	wafv2: ListSettlementRecords
	wafv2: ListWebACLs
	wafv2: PutLoggingConfiguration
	wafv2: PutManagedRuleSetVersions
	WAFv2: UpdateIPSet
	wafv2: UpdateManagedRuleSetVersionExpiryDate
	wafv2: UpdateRegexPatternSet
	wafv2: UpdateRuleGroup
	wafv2: ACL UpdateWeb

Awalan layanan	Tindakan
wellarchitected	wellarchitected: AssociateLenses wellarchitected: AssociateProfiles wellarchitected: CreateLensShare wellarchitected: CreateLensVersion wellarchitected: CreateMilestone wellarchitected: CreateProfile wellarchitected: CreateProfileShare wellarchitected: CreateReviewTemplate wellarchitected: CreateWorkload wellarchitected: CreateWorkloadShare wellarchitected: DeleteLens wellarchitected: DeleteLensShare wellarchitected: DeleteProfile wellarchitected: DeleteProfileShare wellarchitected: DeleteReviewTemplate wellarchitected: DeleteTemplateShare wellarchitected: DeleteWorkload wellarchitected: DeleteWorkloadShare wellarchitected: DisassociateLenses wellarchitected: DisassociateProfiles wellarchitected: ExportLens

Awalan layanan	Tindakan
	wellarchitected: GetAnswer wellarchitected: GetConsolidatedReport wellarchitected: GetGlobalSettings wellarchitected: GetLens wellarchitected: GetLensReview wellarchitected: GetLensReviewReport wellarchitected: GetLensVersionDifference wellarchitected: GetMilestone wellarchitected: GetProfile wellarchitected: GetProfileTemplate wellarchitected: GetReviewTemplate wellarchitected: GetReviewTemplateAnswer wellarchitected: GetReviewTemplateLensReview wellarchitected: GetWorkload wellarchitected: ImportLens wellarchitected: ListAnswers wellarchitected: ListCheckDetails wellarchitected: ListCheckSummaries wellarchitected: ListLensReviewImprovements wellarchitected: ListLensReviews wellarchitected: ListLensShares

Awalan layanan	Tindakan
	wellarchitected: ListLenses wellarchitected: ListMilestones wellarchitected: ListNotifications wellarchitected: ListProfileNotifications wellarchitected: ListProfileShares wellarchitected: ListProfiles wellarchitected: ListReviewTemplateAnswers wellarchitected: ListReviewTemplates wellarchitected: ListShareInvitations wellarchitected: ListTemplateShares wellarchitected: ListWorkloadShares wellarchitected: ListWorkloads wellarchitected: UpdateAnswer wellarchitected: UpdateGlobalSettings wellarchitected: UpdateIntegration wellarchitected: UpdateLensReview wellarchitected: UpdateProfile wellarchitected: UpdateReviewTemplate wellarchitected: UpdateReviewTemplateLensReview wellarchitected: UpdateShareInvitation wellarchitected: UpdateWorkload

Awalan layanan	Tindakan
	wellarchitected: UpdateWorkloadShare wellarchitected: UpgradeLensReview wellarchitected: UpgradeProfileVersion wellarchitected: UpgradeReviewTemplateLensReview

Awalan layanan	Tindakan
kebijaksanaan	kebijaksanaan: CreateAssistant
	kebijaksanaan: CreateAssistantAssociation
	kebijaksanaan: CreateContent
	kebijaksanaan: CreateKnowledgeBase
	kebijaksanaan: CreateQuickResponse
	kebijaksanaan: CreateSession
	kebijaksanaan: DeleteAssistant
	kebijaksanaan: DeleteAssistantAssociation
	kebijaksanaan: DeleteContent
	kebijaksanaan: DeleteImportJob
	kebijaksanaan: DeleteKnowledgeBase
	kebijaksanaan: DeleteQuickResponse
	kebijaksanaan: GetAssistant
	kebijaksanaan: GetAssistantAssociation
	kebijaksanaan: GetContent
	kebijaksanaan: GetContentAssociation
	kebijaksanaan: GetContentSummary
	kebijaksanaan: GetImportJob
	kebijaksanaan: GetKnowledgeBase
	kebijaksanaan: GetRecommendations
	kebijaksanaan: GetSession

Awalan layanan	Tindakan
	kebijaksanaan: ListAssistantAssociations
	kebijaksanaan: ListAssistants
	kebijaksanaan: ListContentAssociations
	kebijaksanaan: ListContents
	kebijaksanaan: ListImportJobs
	kebijaksanaan: ListKnowledgeBases
	kebijaksanaan: ListQuickResponses
	kebijaksanaan: NotifyRecommendationsReceived
	kebijaksanaan: QueryAssistant
	kebijaksanaan: RemoveKnowledgeBaseTemplateUri
	kebijaksanaan: SearchContent
	kebijaksanaan: SearchQuickResponses
	kebijaksanaan: SearchSessions
	kebijaksanaan: StartContentUpload
	kebijaksanaan: StartImportJob
	kebijaksanaan: UpdateContent
	kebijaksanaan: UpdateKnowledgeBaseTemplateUri
	kebijaksanaan: UpdateQuickResponse
	kebijaksanaan: UpdateSession

Awalan layanan	Tindakan
tautan kerja	tautan kerja: AssociateDomain tautan kerja: AssociateWebsiteAuthorizationProvider tautan kerja: AssociateWebsiteCertificateAuthority tautan kerja: CreateFleet tautan kerja: DeleteFleet tautan kerja: DescribeAuditStreamConfiguration tautan kerja: DescribeCompanyNetworkConfiguration tautan kerja: DescribeDevice tautan kerja: DescribeDevicePolicyConfiguration tautan kerja: DescribeDomain tautan kerja: DescribeFleetMetadata tautan kerja: DescribeIdentityProviderConfiguration tautan kerja: DescribeWebsiteCertificateAuthority tautan kerja: DisassociateDomain tautan kerja: DisassociateWebsiteAuthorizationProvider tautan kerja: DisassociateWebsiteCertificateAuthority tautan kerja: ListDevices tautan kerja: ListDomains tautan kerja: ListFleets tautan kerja: ListWebsiteAuthorizationProviders tautan kerja: ListWebsiteCertificateAuthorities

Awalan layanan	Tindakan
	tautan kerja: RestoreDomainAccess tautan kerja: RevokeDomainAccess tautan kerja: SignOutUser tautan kerja: UpdateAuditStreamConfiguration tautan kerja: UpdateCompanyNetworkConfiguration tautan kerja: UpdateDevicePolicyConfiguration tautan kerja: UpdateDomainMetadata tautan kerja: UpdateFleetMetadata tautan kerja: UpdateIdentityProviderConfiguration

Awalan layanan	Tindakan
ruang kerja	ruang kerja: AcceptAccountLinkInvitation
	ruang kerja: AssociateConnectionAlias
	ruang kerja: AssociateIpGroups
	ruang kerja: AssociateWorkspaceApplication
	ruang kerja: CopyWorkspaceImage
	ruang kerja: CreateAccountLinkInvitation
	ruang kerja: CreateConnectClientAddIn
	ruang kerja: CreateConnectionAlias
	ruang kerja: CreateIpGroup
	ruang kerja: CreateStandbyWorkspaces
	ruang kerja: CreateUpdatedWorkspaceImage
	ruang kerja: CreateWorkspaceBundle
	ruang kerja: CreateWorkspaceImage
	ruang kerja: CreateWorkspaces
	ruang kerja: CreateWorkspacesPool
	ruang kerja: DeleteAccountLinkInvitation
	ruang kerja: DeleteClientBranding
	ruang kerja: DeleteConnectClientAddIn
	ruang kerja: DeleteConnectionAlias
	ruang kerja: DeleteIpGroup
	ruang kerja: DeleteWorkspaceBundle

Awalan layanan	Tindakan
	ruang kerja: DeleteWorkspaceImage
	ruang kerja: DeployWorkspaceApplications
	ruang kerja: DeregisterWorkspaceDirectory
	ruang kerja: DescribeAccount
	ruang kerja: DescribeAccountModifications
	ruang kerja: DescribeApplicationAssociations
	ruang kerja: DescribeApplications
	ruang kerja: DescribeBundleAssociations
	ruang kerja: DescribeClientBranding
	ruang kerja: DescribeClientProperties
	ruang kerja: DescribeConnectClientAddIns
	ruang kerja: DescribeConnectionAliasPermissions
	ruang kerja: DescribeConnectionAliases
	ruang kerja: DescribeCustomWorkspaceImageImport
	ruang kerja: DescribeImageAssociations
	ruang kerja: DescribeIpGroups
	ruang kerja: DescribeWorkspaceAssociations
	ruang kerja: DescribeWorkspaceBundles
	ruang kerja: DescribeWorkspaceDirectories
	ruang kerja: DescribeWorkspaceImagePermissions
	ruang kerja: DescribeWorkspaceSnapshots

Awalan layanan	Tindakan
	ruang kerja: DescribeWorkspaces
	ruang kerja: DescribeWorkspacesConnectionStatus
	ruang kerja: DescribeWorkspacesPoolSessions
	ruang kerja: DescribeWorkspacesPools
	ruang kerja: DisassociateConnectionAlias
	ruang kerja: DisassociateIpGroups
	ruang kerja: DisassociateWorkspaceApplication
	ruang kerja: GetAccountLink
	ruang kerja: ImportClientBranding
	ruang kerja: ImportWorkspaceImage
	ruang kerja: ListAccountLinks
	ruang kerja: ListAvailableManagementCidrRanges
	ruang kerja: MigrateWorkspace
	ruang kerja: ModifyAccount
	ruang kerja: ModifyCertificateBasedAuthProperties
	ruang kerja: ModifyClientProperties
	ruang kerja: ModifyEndpointEncryptionMode
	ruang kerja: ModifySamlProperties
	ruang kerja: ModifySelfservicePermissions
	ruang kerja: ModifyStreamingProperties
	ruang kerja: ModifyWorkspaceAccessProperties

Awalan layanan	Tindakan
	ruang kerja: ModifyWorkspaceCreationProperties
	ruang kerja: ModifyWorkspaceProperties
	ruang kerja: ModifyWorkspaceState
	ruang kerja: RebootWorkspaces
	ruang kerja: RebuildWorkspaces
	ruang kerja: RegisterWorkspaceDirectory
	ruang kerja: RejectAccountLinkInvitation
	ruang kerja: RestoreWorkspace
	ruang kerja: StartWorkspaces
	ruang kerja: StartWorkspacesPool
	ruang kerja: StopWorkspaces
	ruang kerja: StopWorkspacesPool
	ruang kerja: TerminateWorkspaces
	ruang kerja: TerminateWorkspacesPool
	ruang kerja: TerminateWorkspacesPoolSession
	ruang kerja: UpdateConnectClientAddIn
	ruang kerja: UpdateConnectionAliasPermission
	ruang kerja: UpdateWorkspaceBundle
	ruang kerja: UpdateWorkspaceImagePermission
	ruang kerja: UpdateWorkspacesPool

Awalan layanan	Tindakan
xray	xray: CreateGroup xray: CreateSamplingRule xray: DeleteGroup xray: DeleteResourcePolicy xray: DeleteSamplingRule xray: GetEncryptionConfig xray: GetGroup xray: GetGroups xray: GetInsight xray: GetInsightEvents xray: GetInsightImpactGraph xray: GetInsightSummaries xray: GetSamplingRules xray: ListResourcePolicies xray: PutEncryptionConfig xray: PutResourcePolicy xray: UpdateGroup xray: UpdateSamplingRule

Kuota Penganalisis Akses IAM

IAM Access Analyzer memiliki kuota berikut:

Sumber daya	Kuota default	Kuota maksimum
Penganalisis tingkat akun maksimum per jenis penganalisis per per Wilayah Akun AWS	1	1
Penganalisis akses eksternal atau tidak terpakai tingkat organisasi maksimum per jenis penganalisis per Wilayah Akun AWS	5	20[1]
Penganalisis akses internal tingkat organisasi maksimum per organisasi per Wilayah AWS	1	1
Aturan arsip maksimum per penganalisis	100 Setiap aturan arsip dapat memiliki hingga 20 nilai per kriteria.	1.000[1]
Jumlah maksimum pratinjau akses per penganalisis per jam	1.000	1.000
AWS CloudTrail file log diproses per generasi kebijakan	100.000	100.000
Pembuatan kebijakan yang bersamaan	1	1
Ukuran AWS CloudTrail data pembuatan kebijakan	25 GB	25 GB
Rentang AWS CloudTrail waktu pembuatan kebijakan	90 hari	90 hari

Sumber daya	Kuota default	Kuota maksimum
Pembuatan kebijakan per hari	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Asia Pasifik (Jakarta): 5 Eropa (Milan): 5 Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50 **Note** Permintaan pembuatan kebijakan yang dibatalkan berlaku untuk kuota harian.	Afrika (Cape Town): 5 Asia Pasifik (Hong Kong): 5 Asia Pasifik (Jakarta): 5 Eropa (Milan): 5 Timur Tengah (Bahrain): 5 Semua wilayah lain yang didukung: 50

Beberapa kuota dapat dikonfigurasi pelanggan menggunakan Service Quotas.

Memecahkan masalah IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah umum saat Anda bekerja dengan AWS Identity and Access Management (IAM).

Masalah

- Saya tidak bisa masuk ke AWS akun
- Saya kehilangan access key saya
- Variabel kebijakan tidak berfungsi
- Perubahan yang saya buat tidak selalu langsung terlihat
- Saya tidak berwenang untuk melakukan: iam: DeleteVirtualMFADevice
- Bagaimana cara membuat pengguna IAM dengan aman?
- Sumber daya tambahan
- Memecahkan masalah pesan kesalahan akses ditolak
- Memecahkan masalah pesan kesalahan akses ditolak dengan ID otorisasi (Pratinjau)
- Memecahkan masalah dengan pengguna root
- Memecahkan masalah kebijakan IAM
- Memecahkan Masalah Kunci Sandi dan Kunci Keamanan FIDO
- Memecahkan masalah peran IAM
- Memecahkan masalah IAM dan Amazon EC2
- Memecahkan masalah IAM dan Amazon S3
- Memecahkan masalah federasi SAML dengan IAM

Saya tidak bisa masuk ke AWS akun

Verifikasi bahwa Anda memiliki kredensial yang benar dan bahwa Anda menggunakan metode yang benar untuk masuk. Untuk informasi selengkapnya, lihat Memecahkan masalah masuk di Panduan AWS Sign-In Pengguna.

Saya kehilangan access key saya

Access key terdiri atas dua bagian:

- Pengidentifikasi kunci akses. Ini bukan rahasia, dan dapat dilihat di konsol IAM di mana pun kunci akses dicantumkan, seperti di laman ringkasan pengguna.
- Kunci akses rahasia. Ini tersedia saat Anda membuat pasangan access key di awal. Seperti kata sandi, kunci ini tidak dapat diambil lagi di lain waktu. Jika Anda kehilangan secret access key, Anda harus membuat pasangan access key baru. Jika Anda sudah memiliki jumlah maksimal access key, Anda harus menghapus pasangan yang ada sebelum Anda dapat membuat yang lain.

Jika Anda kehilangan secret access key, Anda harus menghapus access key dan membuat access key baru. Untuk instruksi lebih lanjut, lihatPerbarui kunci akses.

Variabel kebijakan tidak berfungsi

Jika variabel kebijakan Anda tidak berfungsi, salah satu kesalahan berikut telah terjadi:

Tanggal salah di elemen kebijakan Versi.

Verifikasi bahwa semua kebijakan yang mencakup variabel menyertakan nomor versi berikut dalam kebijakan: `"Version": "2012-10-17"`. Tanpa nomor versi yang benar, variabel tidak diganti selama evaluasi. Alih-alih, variabel dievaluasi secara literal. Kebijakan yang tidak menyertakan variabel masih berfungsi saat Anda menyertakan nomor versi terbaru.

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan dibuat saat Anda mengubah kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola. Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat Elemen kebijakan IAM JSON: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Peningkatan versi IAM”.

Karakter variabel dalam huruf yang salah.

Verifikasi bahwa variabel kebijakan Anda berada di kasus yang tepat. Lihat perinciannya di Elemen kebijakan IAM: Variabel dan tag.

Perubahan yang saya buat tidak selalu langsung terlihat

Sebagai layanan yang diakses melalui komputer di pusat data di seluruh dunia, IAM menggunakan model komputasi terdistribusi yang disebut konsistensi akhir. Setiap perubahan yang Anda buat di

IAM (atau AWS layanan lain), termasuk tag kontrol akses berbasis atribut (ABAC), membutuhkan waktu untuk terlihat dari semua titik akhir yang mungkin. Beberapa penundaan dihasilkan dari waktu yang diperlukan untuk mengirim data dari server ke server, zona replikasi ke zona replikasi, dan Wilayah ke Wilayah. IAM juga menggunakan caching untuk meningkatkan kinerja, tetapi dalam beberapa kasus ini dapat menambah waktu. Perubahan mungkin tidak terlihat sampai waktu data yang disimpan di-cache sebelumnya habis.

Anda harus merancang aplikasi global untuk memperhitungkan kemungkinan penundaan ini. Pastikan aplikasi bekerja sesuai harapan, bahkan ketika perubahan yang dilakukan di satu lokasi tidak secara langsung terlihat di lokasi lain. Perubahan tersebut mencakup membuat atau memperbarui pengguna, kelompok, peran, atau kebijakan. Kami menyarankan agar Anda tidak menyertakan perubahan IAM tersebut di jalur kode ketersediaan tinggi yang kritis dari aplikasi Anda. Sebaliknya, buat perubahan IAM dalam inisialisasi terpisah atau rutinitas pengaturan yang lebih jarang Anda lakukan. Selain itu, pastikan untuk memverifikasi bahwa perubahan telah disebar merata sebelum alur kerja produksi bergantung padanya.

Untuk informasi lebih lanjut tentang bagaimana beberapa AWS layanan lain terpengaruh oleh ini, lihat sumber daya berikut:

- Amazon DynamoDB: Baca konsistensi di Panduan Pengembang DynamoDB, dan Baca Konsistensi di Panduan Pengembang Amazon DynamoDB.
- Amazon EC2: Konsistensi Akhir EC2 dalam Referensi API Amazon EC2.
- Amazon EMR: Memas tikan Konsistensi Saat Menggunakan Amazon S3 dan Amazon EMR untuk Alur Kerja ETL di Blog Big Data AWS
- Amazon Redshift: M engelola Konsistensi Data di Panduan Peng embang Database Amazon Redshift
- Amazon S3: Model Konsistensi Data Amazon S3 dalam Panduan Pengguna Layanan Penyimpanan Sederhana Amazon

Saya tidak berwenang untuk melakukan: iam: DeleteVirtualMFADevice

Anda mungkin menerima kesalahan berikut saat mencoba menugaskan atau menghapus perangkat MFA virtual untuk diri sendiri atau orang lain:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to
 perform: iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego
 with an explicit deny
```

Hal ini dapat terjadi jika sebelumnya seseorang menugaskan perangkat MFA virtual ke pengguna di konsol IAM dan kemudian membatalkan proses. Ini membuat perangkat MFA virtual untuk pengguna di IAM tetapi tidak pernah menetapkannya kepada pengguna. Hapus perangkat MFA virtual yang ada sebelum Anda membuat perangkat MFA virtual baru dengan nama perangkat yang sama.

Untuk mengatasi masalah ini, administrator seharusnya tidak mengedit izin kebijakan. Sebagai gantinya, administrator harus menggunakan AWS API AWS CLI or untuk menghapus perangkat MFA virtual yang ada tetapi tidak ditetapkan.

Untuk menghapus perangkat MFA virtual yang ada tetapi tidak ditetapkan

1. Lihat perangkat MFA virtual di akun Anda.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API: `ListVirtualMFADevices`
2. Dalam tanggapan, cari ARN perangkat MFA virtual untuk pengguna yang Anda coba perbaiki.
3. Hapus perangkat MFA virtual.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API: `DeleteVirtualMFADevice`

Bagaimana cara membuat pengguna IAM dengan aman?

Jika Anda memiliki karyawan yang memerlukan akses ke AWS, Anda dapat memilih untuk membuat pengguna IAM atau menggunakan Pusat Identitas IAM untuk otentikasi. Jika Anda menggunakan IAM, sarankan AWS Anda membuat pengguna IAM dan mengkomunikasikan kredensialnya dengan aman kepada karyawan. Jika Anda tidak berada di sebelah karyawan Anda secara fisik, gunakan alur kerja yang aman untuk memberi tahu kredensialnya kepada karyawan.

Gunakan alur kerja aman berikut untuk membuat pengguna baru di IAM:

1. Buat pengguna baru menggunakan Konsol Manajemen AWS. Pilih untuk memberikan Konsol Manajemen AWS akses dengan kata sandi yang dihasilkan. Jika perlu, pilih kotak centang Users must create a new password at next sign-in(Pengguna harus membuat kata sandi baru saat

masuk berikutnya). Jangan tambahkan kebijakan izin ke pengguna sampai setelah mereka mengubah sandi mereka.

2. Setelah pengguna ditambahkan, salin URL masuk, nama pengguna, dan sandi untuk pengguna baru. Untuk melihat kata sandi, pilih Show (Tampilkan).
3. Kirim kata sandi ke karyawan Anda menggunakan metode komunikasi yang aman di perusahaan Anda, seperti email, obrolan, atau sistem tiket. Secara terpisah, berikan pengguna Anda tautan konsol pengguna IAM dan nama pengguna mereka. Beri tahu karyawan untuk mengonfirmasi bahwa mereka dapat berhasil masuk sebelum Anda memberikan izin kepada karyawan tersebut.
4. Setelah karyawan mengonfirmasi, tambahkan izin yang mereka perlukan. Sebagai praktik terbaik keamanan, tambahkan kebijakan yang mengharuskan pengguna mengautentikasi menggunakan MFA untuk mengelola kredensialnya. Untuk contoh kebijakan, lihat AWS: Memungkinkan pengguna IAM yang diautentikasi MFA untuk mengelola kredensialnya sendiri di halaman kredensi Keamanan.

Sumber daya tambahan

Sumber daya berikut dapat membantu Anda memecahkan masalah saat Anda bekerja AWS.

- AWS CloudTrail Panduan Pengguna — Gunakan AWS CloudTrail untuk melacak riwayat panggilan API yang dilakukan AWS dan menyimpan informasi tersebut dalam file log. Ini membantu Anda menentukan pengguna dan akun mana yang mengakses sumber daya di akun Anda, kapan panggilan dilakukan, tindakan apa yang diminta, dan banyak lagi. Untuk informasi selengkapnya, lihat Pencatatan IAM dan AWS STS Panggilan API dengan AWS CloudTrail.
- AWS Pusat Peng etahuan — Temukan FAQ dan tautan ke sumber daya lain untuk membantu Anda memecahkan masalah.
- AWS Pusat Dukungan — Dapatkan dukungan teknis.
- AWS Pusat Dukungan Premium — Dapatkan dukungan teknis premium.

Memecahkan masalah pesan kesalahan akses ditolak

Informasi berikut dapat membantu Anda mengidentifikasi, mendiagnosis, dan menyelesaikan kesalahan akses yang ditolak dengan AWS Identity and Access Management. Kesalahan akses ditolak muncul ketika AWS secara eksplisit atau implisit menolak permintaan otorisasi.

- Penolakan eksplisit terjadi ketika kebijakan berisi `Deny` pernyataan untuk AWS tindakan tertentu.

- Penolakan implisit terjadi ketika tidak ada `Deny` pernyataan yang berlaku dan juga tidak ada `Allow` pernyataan yang berlaku. Karena kebijakan IAM menolak prinsipal IAM secara default, kebijakan harus secara eksplisit mengizinkan prinsipal untuk melakukan tindakan. Jika tidak, kebijakan secara implisit menolak akses. Untuk informasi selengkapnya, lihat Perbedaan antara penolakan tegas dan implisit..

Saat Anda membuat permintaan ke layanan atau sumber daya, beberapa kebijakan mungkin berlaku untuk permintaan tersebut. Tinjau semua kebijakan yang berlaku selain kebijakan yang ditentukan dalam pesan kesalahan.

- Jika beberapa kebijakan dari jenis kebijakan yang sama menolak permintaan, pesan kesalahan akses ditolak tidak menentukan jumlah kebijakan yang dievaluasi.
- Jika beberapa jenis kebijakan menolak permintaan otorisasi, AWS sertakan hanya satu dari jenis kebijakan tersebut dalam pesan kesalahan.

> **Important**
>
> Kesulitan masuk ke AWS? Pastikan Anda berada di halaman masuk AWS yang benar untuk jenis pengguna Anda. Jika Anda adalah Pengguna root akun AWS (pemilik akun), Anda dapat masuk AWS menggunakan kredensional yang Anda atur saat membuat Akun AWS. Jika Anda adalah pengguna IAM, administrator akun Anda dapat memberi Anda kredentif AWS masuk. Jika Anda perlu meminta dukungan, jangan gunakan tautan umpan balik di halaman ini. Formulir diterima oleh tim AWS Dokumentasi, tidak Dukungan. Sebagai gantinya, pada halaman Hu bungi Kami pilih Masih tidak dapat masuk ke AWS akun Anda dan kemudian pilih salah satu opsi dukungan yang tersedia.

Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan

- Periksa apakah pesan kesalahan menyertakan jenis dan Nama Sumber Daya Amazon (ARN) dari kebijakan yang bertanggung jawab untuk menolak akses. Jika ini masalahnya, maka periksa pernyataan tolak untuk tindakan dalam kebijakan yang ditentukan. Jika jenis kebijakan disediakan tetapi tidak ada ARN kebijakan, maka fokuslah pada masalah pemecahan masalah untuk jenis kebijakan tersebut: Periksa pernyataan penolakan untuk tindakan dalam kebijakan jenis yang ditentukan. Jika pesan kesalahan tidak menyebutkan jenis kebijakan yang bertanggung jawab

untuk menolak akses, gunakan panduan lainnya di bagian ini untuk memecahkan masalah lebih lanjut.

- Pastikan bahwa Anda memiliki izin kebijakan berbasis identitas untuk memanggil tindakan dan sumber daya yang Anda minta. Jika ada kondisi yang ditetapkan, Anda juga harus memenuhi kondisi tersebut saat mengirimkan permintaan. Untuk informasi tentang melihat atau mengubah kebijakan untuk pengguna, grup, atau peran IAM, lihat Kelola kebijakan IAM.
- Jika Konsol Manajemen AWS mengembalikan pesan yang menyatakan bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk mendapatkan bantuan. Administrator memberi Anda kredensial masuk atau tautan masuk.

  Contoh kesalahan berikut terjadi ketika pengguna `mateojackson` IAM mencoba menggunakan konsol untuk melihat detail tentang *`my-example-widget`* sumber daya fiksi tetapi tidak memiliki `widgets:`*`GetWidget`* izin fiksi.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  Dalam hal ini, Mateo harus meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan mengakses ke sumber daya *`my-example-widget`* menggunakan tindakan `widgets:`*`GetWidget`*.
- Apakah Anda mencoba mengakses layanan yang mendukung kebijakan berbasis sumber daya, seperti Amazon S3, Amazon SNS, atau Amazon SQS? Jika ya, verifikasikan bahwa kebijakan tersebut menetapkan Anda sebagai penanggung jawab dan memberi Anda akses. Jika Anda mengajukan permintaan ke layanan di dalam akun Anda, kebijakan berbasis identitas atau kebijakan berbasis sumber daya dapat memberikan izin kepada Anda. Jika Anda membuat permintaan layanan di akun yang berbeda, maka kebijakan berbasis identitas dan kebijakan berbasis sumber daya Anda harus memberi Anda izin. Untuk melihat layanan yang mendukung kebijakan berbasis sumber daya, lihat AWS layanan yang bekerja dengan IAM.
- Jika kebijakan Anda menyertakan kondisi dengan pasangan nilai kunci, tinjau dengan cermat. Contohnya termasuk kunci kondisi `aws:RequestTag/`*`tag-key`* global, kunci kondisi AWS KMS `kms:EncryptionContext:`*`encryption_context_key`*, dan kunci `ResourceTag/`*`tag-key`* kondisi yang didukung oleh beberapa layanan. Pastikan bahwa nama kunci tidak cocok dengan beberapa hasil. Karena nama kunci kondisi tidak peka huruf besar/kecil, kondisi yang memeriksa kunci bernama `foo` cocok dengan `foo`, `Foo`, atau `F00`. Jika permintaan Anda menyertakan beberapa pasangan kunci-nilai dengan nama kunci yang berbeda hanya berdasarkan kasus, maka

akses Anda mungkin ditolak secara tidak terduga. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM JSON: Kondisi.

- Jika Anda memiliki batas izin, verifikasikan bahwa kebijakan yang digunakan untuk batas izin memungkinkan permintaan Anda. Jika kebijakan berbasis identitas Anda mengizinkan permintaan, tetapi batas izin Anda tidak, maka permintaan ditolak. Batas izin mengontrol izin maksimum yang dapat dimiliki oleh prinsipal IAM (pengguna atau peran). Resource-based kebijakan tidak dibatasi oleh batas izin. Batas izin bukanlah hal yang umum. Untuk informasi selengkapnya tentang cara AWS mengevaluasi kebijakan, lihatLogika evaluasi kebijakan.
- Jika Anda menandatangani permintaan secara manual (tanpa menggunakan SDK AWS), verifikasi bahwa Anda sudah menandatangani permintaan dengan benar.
- Jika Anda menggunakan kebijakan titik akhir Amazon VPC dan Anda mendapatkan kesalahan akses ditolak yang tidak masuk AWS CloudTrail, itu mungkin karena akun pemilik titik akhir VPC berbeda dari akun panggilan atau akun peran target.

Saya mendapatkan “akses ditolak” ketika saya membuat permintaan dengan kredensial keamanan sementara

- Pertama, pastikan bahwa akses Anda tidak ditolak karena alasan yang tidak terkait dengan kredensial sementara Anda. Untuk informasi selengkapnya, lihat Saya mendapatkan “akses ditolak” ketika saya mengajukan permintaan ke AWS layanan.
- Verifikasi bahwa layanan menerima kredensial keamanan sementara, lihat AWS layanan yang bekerja dengan IAM.
- Verifikasi bahwa permintaan Anda ditandatangani dengan benar dan bahwa permintaan tersebut memiliki bentuk yang baik. Untuk detailnya, lihat dokumentasi toolkit atau Gunakan kredenSIAL sementara dengan AWS sumber daya Anda.
- Verifikasikan bahwa kredensial keamanan sementara Anda belum kedaluwarsa. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.
- Verifikasi bahwa pengguna atau peran IAM memiliki izin yang benar. Izin untuk kredensial keamanan sementara berasal dari pengguna atau peran IAM. Akibatnya, izin dibatasi untuk mereka yang diberikan peran yang kredensial sementaranya telah Anda gunakan. Untuk informasi selengkapnya tentang bagaimana izin untuk kredensial keamanan sementara ditentukan, lihat Izin untuk kredensial keamanan sementara.
- Jika Anda mengambil peran, sesi peran Anda mungkin dibatasi oleh kebijakan sesi. Bila Anda meminta kredentif keamanan sementara secara terprogram menggunakan AWS STS, Anda

dapat meneruskan kebijakan sesi sebaris atau terkelola secara opsional. Kebijakan sesi adalah kebijakan lanjutan yang Anda sampaikan sebagai parameter saat Anda secara terprogram membuat sesi kredensial sementara untuk peran. Anda dapat menyampaikan satu dokumen kebijakan sesi inline JSON menggunakan parameter `Policy`. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas dan kebijakan sesi peran. Selain itu, jika administrator Anda atau program khusus memberi Anda kredensial sementara, mereka mungkin telah menyertakan kebijakan sesi untuk membatasi akses Anda.

- Jika Anda adalah kepala AWS STS sekolah pengguna federasi, sesi Anda mungkin dibatasi oleh kebijakan sesi. Anda membuat sesi pengguna federasi dengan masuk AWS sebagai pengguna IAM dan kemudian meminta token federasi. Untuk informasi selengkapnya, lihat Meminta kredenSIAL melalui broker identitas khusus. Jika Anda atau pialang identitas Anda menyampaikan kebijakan sesi sambil meminta token federasi, maka sesi Anda akan dibatasi oleh kebijakan tersebut. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas pengguna IAM dan kebijakan sesi. Untuk informasi selengkapnya tentang kebijakan sesi, lihat Kebijakan sesi.
- Jika Anda mengakses sumber daya yang memiliki kebijakan berbasis sumber daya dengan menggunakan peran, verifikasi bahwa kebijakan tersebut memberikan izin untuk peran tersebut. Misalnya, kebijakan berikut memungkinkan `MyRole` dari akun `111122223333` untuk mengakses `amzn-s3-demo-bucket`.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
  }]
}
```

Contoh pesan kesalahan akses ditolak

Sebagian besar pesan kesalahan akses ditolak muncul dalam format`User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. Dalam contoh ini, *`user`* adalah ARN dari prinsipal yang ditolak akses, *`action`* adalah tindakan layanan yang ditolak

kebijakan, dan *resource* merupakan ARN dari sumber daya tempat kebijakan bertindak. Bid *context* ang ini menyediakan konteks tambahan tentang jenis kebijakan yang menolak akses. Dalam beberapa kasus, itu juga berisi ARN dari kebijakan yang menolak akses.

Jika kebijakan secara eksplisit menolak akses karena kebijakan berisi Deny pernyataan, maka ser AWS takan frasa tersebut with an explicit deny in a *type* policy dalam pesan kesalahan akses ditolak. Frasa ini juga dapat menentukan ARN kebijakan, sebagai berikut:with an explicit deny in a *type* policy: *policy ARN*.

Jika kebijakan secara implisit menolak akses, maka ser AWS takan frasa tersebut because no *type* policy allows the *action* action dalam pesan kesalahan akses ditolak.

> **Note**
>
> Beberapa AWS layanan tidak mendukung format pesan kesalahan akses ditolak ini. Konten pesan kesalahan akses ditolak dapat bervariasi tergantung pada layanan yang membuat permintaan otorisasi.

Contoh berikut menunjukkan format untuk berbagai jenis pesan kesalahan akses ditolak.

Akses ditolak karena kebijakan kontrol layanan — penolakan implisit

1. Periksa Allow pernyataan yang hilang untuk tindakan dalam kebijakan kontrol layanan (SCP) Anda. Untuk contoh berikut, tindakannya adalahcodecommit:ListRepositories.
2. Perbarui SCP Anda dengan menambahkan Allow pernyataan. Untuk informasi selengkapnya, lihat Memperbarui SCP di Panduan AWS Organizations Pengguna.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no service control policy allows the codecommit:ListRepositories action
```

Akses ditolak karena kebijakan kontrol layanan — penolakan eksplisit

1. Jika kebijakan ARN disediakan dalam pesan kesalahan, periksa Deny pernyataan untuk tindakan dalam kebijakan kontrol layanan (SCP) yang ditentukan. Dalam contoh di bawah ini, tindakannya adalahcodecommit:ListRepositories.

2. Jika tidak ada kebijakan ARN yang disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan di SCP Anda.
3. Perbarui SCP Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Memper barui kebijakan kontrol layanan (SCP) di Panduan AWS Organizations Pengguna.

Pesan kesalahan dengan kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/service_control_policy/p-
examplepolicyid123
```

Pesan kesalahan tanpa kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy
```

Akses ditolak karena kebijakan kontrol sumber daya — penolakan eksplisit

1. Jika kebijakan ARN disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan dalam kebijakan kontrol sumber daya (RCP) yang ditentukan. Dalam contoh di bawah ini, tindakannya adalah`secretsmanager:GetSecretValue`.
2. Jika tidak ada kebijakan ARN yang disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan di RCP Anda.
3. Perbarui RCP Anda dengan menghapus `Deny` pernyataan. Untuk informasi selengkapnya, lihat Memper barui kebijakan kontrol sumber daya (RCP) di Panduan AWS Organizations Pengguna.

Pesan kesalahan dengan kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/resource_control_policy/p-
examplepolicyid456
```

Pesan kesalahan tanpa kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy
```

Akses ditolak karena kebijakan titik akhir VPC — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan titik akhir Virtual Private Cloud (VPC) Anda. Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui kebijakan titik akhir VPC Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Memper barui kebijakan titik akhir VPC di AWS PrivateLink Panduan.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no VPC endpoint policy allows the codecommit:ListRepositories action
```

Akses ditolak karena kebijakan titik akhir VPC — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan titik akhir Virtual Private Cloud (VPC) Anda. Untuk contoh berikut, tindakannya adalah`codedeploy:ListDeployments`.
2. Perbarui kebijakan titik akhir VPC Anda dengan menghapus `Deny` pernyataan tersebut. Untuk informasi selengkapnya, lihat Memper barui kebijakan titik akhir VPC di AWS PrivateLink Panduan.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a VPC endpoint policy
```

Akses ditolak karena batas izin — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan di batas izin Anda. Untuk contoh berikut, tindakannya adalah`codedeploy:ListDeployments`.

2. Perbarui batas izin Anda dengan menambahkan `Allow` pernyataan ke kebijakan IAM Anda. Untuk informasi selengkapnya, lihat Batas izin untuk entitas IAM dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
because no permissions boundary allows the codedeploy:ListDeployments action
```

Akses ditolak karena batas izin — penolakan eksplisit

1. Jika kebijakan ARN disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan di batas izin yang ditentukan. Dalam contoh di bawah ini, tindakannya adalah`sagemaker:ListModels`.
2. Jika tidak ada kebijakan ARN yang disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan di batas izin yang dilampirkan ke prinsipal.
3. Perbarui batas izin Anda dengan menghapus `Deny` pernyataan dari kebijakan IAM Anda. Untuk informasi selengkapnya, lihat Batas izin untuk entitas IAM dan Edit kebijakan IAM.

Pesan kesalahan dengan kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary: arn:aws:iam::123456789012:policy/
DeveloperPermissionBoundary
```

Pesan kesalahan tanpa kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary
```

Akses ditolak karena kebijakan sesi — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan sesi Anda. Untuk contoh berikut, tindakannya adalah`codecommit:ListRepositories`.
2. Perbarui kebijakan sesi Anda dengan menambahkan `Allow` pernyataan. Untuk informasi selengkapnya, lihat Kebijakan sesi danEdit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no session policy allows the codecommit:ListRepositories action
```

Akses ditolak karena kebijakan sesi — penolakan eksplisit

1. Jika kebijakan ARN disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan dalam kebijakan sesi yang ditentukan. Dalam contoh di bawah ini, tindakannya adalah`codedeploy:ListDeployments`.
2. Jika tidak ada kebijakan ARN yang disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan dalam kebijakan sesi Anda.
3. Perbarui kebijakan sesi Anda dengan menghapus `Deny` pernyataan tersebut. Untuk informasi selengkapnya, lihat Kebijakan sesi dan Edit kebijakan IAM.

Pesan kesalahan dengan kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy: arn:aws:iam::123456789012:policy/
DeveloperSessionPolicy
```

Pesan kesalahan tanpa kebijakan ARN:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy
```

Akses ditolak karena kebijakan berbasis sumber daya — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan berbasis sumber daya Anda. Untuk contoh berikut, tindakannya adalah`secretsmanager:GetSecretValue`.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi lebih lanjut, lihat Resource-based kebijakan dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
because no resource-based policy allows the secretsmanager:GetSecretValue action
```

Akses ditolak karena kebijakan berbasis sumber daya — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan berbasis sumber daya Anda. Untuk contoh berikut, tindakannya adalah`secretsmanager:GetSecretValue`.
2. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi lebih lanjut, lihat Resource-based kebijakan dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource-based policy
```

Akses ditolak karena kebijakan kepercayaan peran — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan kepercayaan peran Anda. Untuk contoh berikut, tindakannya adalah`sts:AssumeRole`.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi lebih lanjut, lihat Resource-based kebijakan dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
because no role trust policy allows the sts:AssumeRole action
```

Akses ditolak karena kebijakan kepercayaan peran — penolakan eksplisit

1. Periksa `Deny` pernyataan eksplisit untuk tindakan dalam kebijakan kepercayaan peran Anda. Untuk contoh berikut, tindakannya adalah`sts:AssumeRole`.
2. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi lebih lanjut, lihat Resource-based kebijakan dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
with an explicit deny in the role trust policy
```

Akses ditolak karena kebijakan berbasis identitas — penolakan implisit

1. Periksa `Allow` pernyataan yang hilang untuk tindakan dalam kebijakan berbasis identitas yang dilampirkan pada identitas. Untuk contoh berikut, tindakan dilamp `codecommit:ListRepositories` irkan pada peranHR.
2. Perbarui kebijakan Anda dengan menambahkan `Allow` pernyataan. Untuk informasi lebih lanjut, lihat Identity-based kebijakan dan Edit kebijakan IAM.

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codecommit:ListRepositories
because no identity-based policy allows the codecommit:ListRepositories action
```

Akses ditolak karena kebijakan berbasis identitas — penolakan eksplisit

1. Jika kebijakan ARN disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan dalam kebijakan yang ditentukan. Dalam contoh di bawah ini, tindakannya adalah`codedeploy:ListDeployments`.
2. Jika tidak ada kebijakan ARN yang disediakan dalam pesan kesalahan, periksa `Deny` pernyataan untuk tindakan dalam kebijakan berbasis identitas yang dilampirkan pada identitas.
3. Perbarui kebijakan Anda dengan menghapus `Deny` pernyataan. Untuk informasi lebih lanjut, lihat Identity-based kebijakan dan Edit kebijakan IAM.

Pesan kesalahan dengan kebijakan ARN:

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
HRAccessPolicy
```

Pesan kesalahan tanpa kebijakan ARN:

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy
```

Memecahkan masalah pesan kesalahan akses ditolak dengan ID otorisasi (Pratinjau)

> **Important**
>
> Access Troubleshooter ada di pratinjau publik.

Gambaran umum

Ketika AWS menolak permintaan untuk API yang didukung, respons kesalahan akses ditolak biasanya menyertakan ID otorisasi. ID ini merujuk evaluasi otorisasi untuk permintaan yang ditolak itu. Untuk mengambil rincian evaluasi dan mengidentifikasi apa yang menyebabkan penolakan, buat permintaan ke `GetRequestAuthorizationDetails` API dan berikan ID otorisasi, atau buka tautan yang ada dalam pesan kesalahan. Ini memungkinkan Anda mengidentifikasi penyebab permintaan yang ditolak dalam satu panggilan, daripada menyelesaikan penolakan satu per satu.

> **Note**
>
> AWS tidak menjamin ID otorisasi untuk setiap permintaan yang ditolak. Lihat Saat Anda tidak menerima ID otorisasi dalam dokumen ini.

Contoh berikut menunjukkan pesan kesalahan akses ditolak yang mencakup kebijakan ARN, tautan ke Konsol AWS Manajemen, dan ID otorisasi:

```
User: arn:aws:iam::123456789012:user/janedoe is not authorized to perform:
 iam:CreateInstanceProfile
on resource: arn:aws:iam::123456789012:instance-profile/my-instance-profile
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
ExampleDenyPolicy.
Go to https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/
authorization-details/EXAMPLE1a2b3c4d5e6f7g8h9i0j for complete details,
or call the GetRequestAuthorizationDetails API with the following authorization id:
 EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

Konsep dan definisi

- Rincian otorisasi. Rincian otorisasi mencakup konteks permintaan, evaluasi yang dilakukan, dan kebijakan yang dievaluasi.
- ID otorisasi. Pengidentifikasi unik yang disertakan dalam respons kesalahan ditolak akses yang didukung. Ini merujuk evaluasi otorisasi untuk permintaan yang ditolak itu, dan merupakan nilai yang Anda berikan ke tautan Konsol AWS Manajemen atau `GetRequestAuthorizationDetails` API.
- Minta konteks. Nilai kunci konteks kondisi dievaluasi untuk permintaan. Ini termasuk kunci konteks kondisi global dan kunci konteks kondisi khusus layanan.
- Evaluasi. Untuk setiap action/resource pasangan dalam permintaan, kunci konteks kondisi yang diterapkan dan hasil yang dievaluasi:`ALLOW`, `EXPLICIT_DENY` (kebijakan secara eksplisit menolak tindakan), atau `IMPLICIT_DENY` (tidak ada kebijakan yang mengizinkan tindakan), dan referensi ke setiap kebijakan dan pernyataan yang cocok dengan action/resource pasangan. Setiap referensi adalah kebijakan ARN (untuk kebijakan terkelola) atau URI (untuk kebijakan sebaris dan kebijakan non-terkelola lainnya) yang digunakan kembali selama respons.
- Kebijakan dievaluasi selama otorisasi. Kebijakan yang AWS dipertimbangkan untuk permintaan, termasuk kebijakan yang tidak cocok dengan action/resource pasangan. Setiap entri memberikan jenis kebijakan (misalnya, berbasis identitas, berbasis sumber daya, SCP, RCP, batas izin, atau kebijakan sesi) dan ARN (untuk kebijakan terkelola) atau URI yang sama yang digunakan dalam evaluasi (untuk kebijakan sebaris dan kebijakan non-terkelola lainnya). Gunakan ARN atau URI untuk mencocokkan kebijakan kembali dengan evaluasi yang mereferensikannya. Bidang “Terlampir ke” menunjukkan ARN utama dari kepala sekolah yang dilampirkan kebijakan yang dievaluasi.

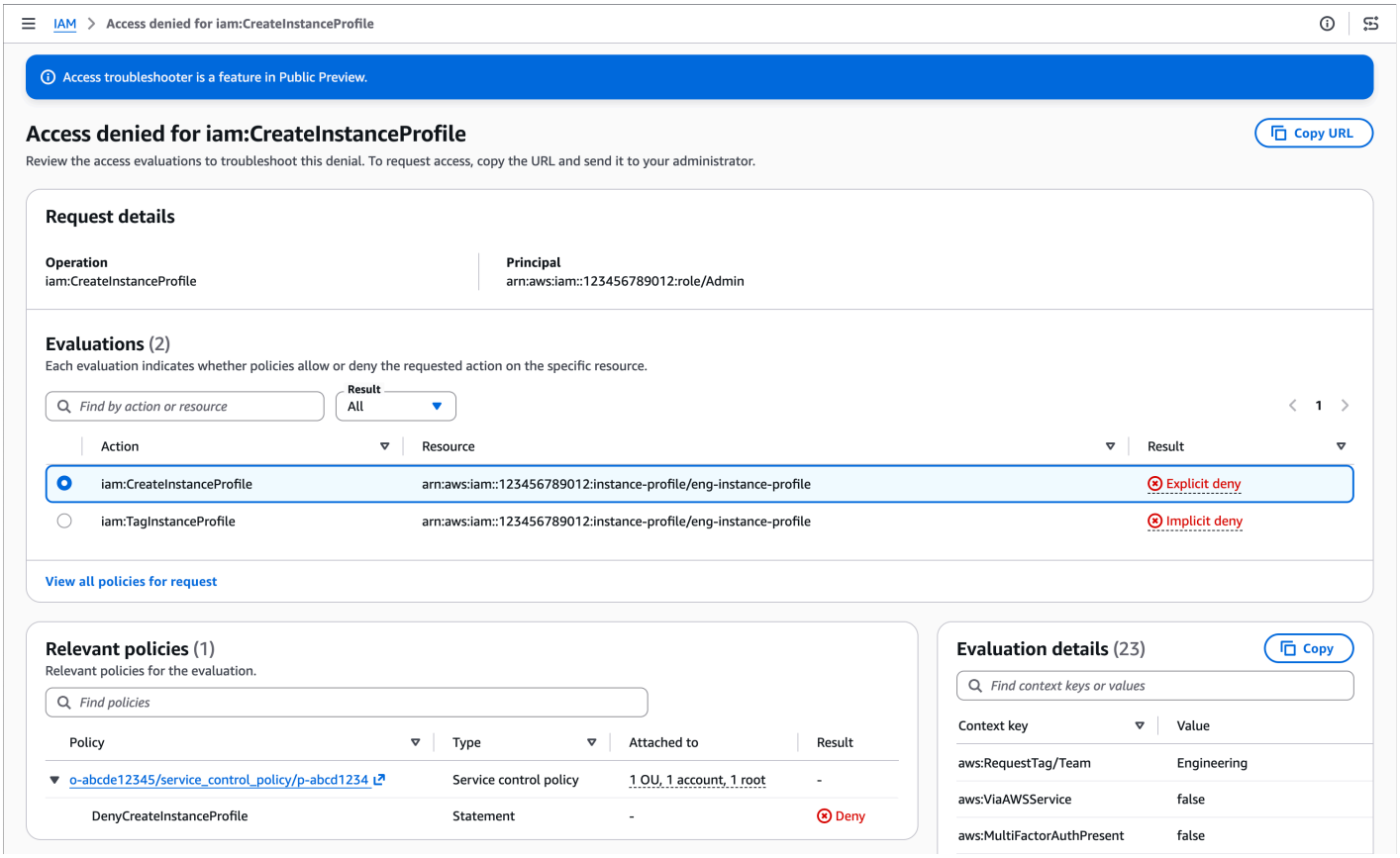
Gunakan hasil ini bersama-sama untuk mengidentifikasi kebijakan mana yang ditolak akses dan mengapa. Evaluasi mencerminkan kebijakan seperti pada saat penolakan. Kebijakan mungkin telah berubah sejak saat itu.

AWS secara otomatis menyertakan ID otorisasi dalam respons kesalahan akses ditolak untuk API yang didukung, tetapi ID otorisasi tidak dijamin. Dalam beberapa kasus, AWS tidak merekam detail otorisasi untuk penolakan, dan respons kesalahan tidak menyertakan ID otorisasi. Ketika detail direkam, mereka mungkin tidak segera tersedia, karena AWS merekamnya secara asinkron. Jika panggilan untuk `GetRequestAuthorizationDetails` mengembalikan respons yang tidak ditemukan, tunggu sebentar dan coba lagi. AWS menyimpan rincian otorisasi selama 24 jam setelah penolakan.

Memecahkan masalah menggunakan AWS Konsol Manajemen

Pilih tautan dalam pesan kesalahan akses ditolak untuk membuka detail otorisasi di Konsol Pemecah Masalah Akses.

- Apa yang harus dicari. Konsol Pemecah Masalah Akses menunjukkan evaluasi untuk setiap action/resource pasangan dalam permintaan, hasil masing-masing (`Allow`,`Explicit deny`, atau`Implicit deny`), dan kebijakan yang dipertimbangkan. Temukan pasangan yang ditolak dan kebijakan yang menyebabkan penolakan. Lihat screenshot berikut sebagai contoh:



- Tentukan apakah penolakan itu dimaksudkan. Tinjau evaluasi sebelum mengubah kebijakan. Penolakan bisa benar, dan penyebabnya terkadang adalah konteks permintaan daripada kebijakan. Misalnya, permintaan mungkin kehilangan tag sesi yang diharapkan atau berasal dari jalur jaringan yang tidak terduga.
- Bagaimana mengatasi penolakan. Temukan kebijakan yang sesuai dan perbarui untuk menyelesaikan masalah akses. Untuk penolakan eksplisit, hapus atau persempit `Deny` pernyataan untuk tindakan tersebut. Untuk penolakan implisit, tambahkan `Allow` pernyataan untuk tindakan tersebut. Untuk informasi selengkapnya, lihat Edit kebijakan IAM.

Siapa yang dapat mengakses detail otorisasi. Untuk permintaan dalam satu akun atau organisasi tunggal, siapa pun yang diberikan akses `iam:GetRequestAuthorizationDetails` dapat mengakses detail permintaan. Untuk permintaan yang mencakup lebih dari satu organisasi, pengguna dalam organisasi tertentu hanya dapat melihat detail yang berkaitan dengan organisasi mereka sendiri.

Memecahkan masalah menggunakan AWS API atau AWS CLI

Untuk memecahkan masalah penolakan menggunakan AWS API atau AWS CLI:

1. Temukan ID otorisasi dalam respons kesalahan akses ditolak.
2. Panggil `GetRequestAuthorizationDetails` dengan ID otorisasi, di tempat yang sama Wilayah AWS di mana penolakan terjadi. Contoh berikut menunjukkan AWS CLI perintah:

```
aws iam-toolbox get-request-authorization-details \
    --region us-east-1 \
    --authorization-id EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

3. Tinjau hasil evaluasi untuk mengidentifikasi kebijakan mana yang menolak akses dan alasan penolakan.
4. Perbarui kebijakan yang diidentifikasi untuk menyelesaikan masalah akses. Untuk informasi selengkapnya, lihat Edit kebijakan IAM.

Saat Anda tidak menerima ID otorisasi

AWS tidak mengembalikan ID otorisasi dalam kasus berikut:

- Otorisasi terlalu besar untuk dicatat. Ketika permintaan melibatkan sejumlah besar kebijakan, AWS dapat memutuskan untuk tidak merekam permintaan tersebut.
- Layanan ini belum didukung. Lihat Layanan yang didukung.
- API tidak didukung. Beberapa API dalam layanan yang didukung tidak mengembalikan ID otorisasi.
- Jenis permintaan tidak didukung. Ini termasuk permintaan yang dibuat oleh layanan atas nama Anda dan permintaan anonim.

Jika Anda tidak menerima ID otorisasi, lihat cara lain Memecahkan masalah pesan kesalahan akses ditolak untuk mendiagnosis penolakan.

Pertimbangan tambahan

Izin yang diperlukan. Untuk memang `GetRequestAuthorizationDetails` gil, Anda harus memiliki izin untuk `iam:GetRequestAuthorizationDetails` tindakan tersebut.

API regional. `GetRequestAuthorizationDetails`adalah API regional. Sebut saja di Wilayah AWS tempat yang sama di mana kesalahan akses ditolak dan ID otorisasi dibuat.

Cross-organization permintaan. Ketika permintaan yang ditolak melibatkan lebih dari satu organisasi, hanya `GetRequestAuthorizationDetails` mengembalikan detail milik organisasi Anda sendiri. Ini berlaku di kedua arah: Jika Anda berada di organisasi penelepon yang ditolak, Anda hanya melihat detail yang berkaitan dengan organisasi itu. Jika Anda berada di organisasi sumber daya yang ditolak, Anda hanya melihat detail yang berkaitan dengan organisasi itu. Detail yang dimiliki oleh organisasi lain, seperti kebijakan organisasi atau nilai kunci konteks, tidak dikembalikan.

Jika Anda mendapatkan kesalahan akses ditolak dari API, layanan, atau wilayah yang belum menyertakan ID otorisasi, lihatMemecahkan masalah pesan kesalahan akses ditolak.

Layanan yang didukung

Dukungan untuk `GetRequestAuthorizationDetails` diluncurkan secara bertahap di seluruh AWS API dan Wilayah AWS. Dukungan layanan berikut`GetRequestAuthorizationDetails`:

- Sebagian besar API di IAM

Memecahkan masalah dengan pengguna root

Gunakan informasi di sini untuk membantu Anda memecahkan masalah yang terkait dengan pengguna root. Akun AWS

> **Note**
>
> Akun AWS dikelola menggunakan AWS Organizations mungkin memiliki akses root terpusat yang diaktifkan untuk akun anggota. Akun anggota ini tidak memiliki kredensi pengguna root, tidak dapat masuk sebagai pengguna root, dan dicegah memulihkan kata sandi pengguna

root. Hubungi administrator Anda jika Anda perlu melakukan tugas yang memerlukan kredensi pengguna root.

Saya tidak dapat melakukan tugas yang saya harapkan dapat dilakukan saat masuk sebagai pengguna root akun

Akun Anda mungkin anggota organisasi di AWS Organizations. Administrator organisasi Anda mungkin memiliki kebijakan kontrol layanan (SCP) untuk membatasi izin akun Anda. SCPs berdampak pada semua pengguna, termasuk pengguna root. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam Panduan Pengguna AWS Organizations .

Saya lupa kata sandi pengguna root untuk saya Akun AWS

Jika Anda adalah pengguna root dan Anda telah kehilangan atau lupa kata sandi untuk Anda Akun AWS, Anda dapat mengatur ulang kata sandi Anda. Anda harus mengetahui alamat email yang digunakan untuk membuat Akun AWS, dan Anda harus memiliki akses ke akun email. Untuk informasi selengkapnya, lihat Setel ulang kata sandi pengguna root yang hilang atau terlupakan.

Saya tidak memiliki akses ke email untuk saya Akun AWS

Saat Anda membuat Akun AWS, Anda memberikan alamat email dan kata sandi. Ini adalah kredensial untuk Pengguna root akun AWS. Jika Anda tidak yakin dengan alamat email yang terkait dengan Anda Akun AWS, cari pesan yang dikirim dari `@signin.aws` atau `@verify.signin.aws` ke alamat email untuk organisasi Anda yang mungkin telah digunakan untuk membuka Akun AWS.

Jika Anda tahu alamat email tetapi tidak lagi memiliki akses ke email, cobalah untuk memulihkan akses ke email. Gunakan salah satu opsi berikut untuk mendapatkan kembali akses ke email Anda:

- Jika Anda memiliki domain untuk alamat email, Anda dapat mengembalikan alamat email yang dihapus. Atau, Anda dapat mengatur tangkapan semua untuk akun email Anda. Catch-all mengumpulkan semua pesan yang dikirim ke alamat email yang tidak lagi ada di server email dan mengarahkannya ke alamat email lain.
- Jika alamat email pada akun adalah bagian dari sistem email perusahaan Anda, kami sarankan untuk menghubungi administrator sistem IT Anda. Mereka mungkin dapat membantu Anda mendapatkan akses ke email tersebut.

Jika Anda masih tidak dapat masuk Akun AWS, Anda dapat menemukan opsi dukungan alternatif di Hubungi kami.

Memecahkan masalah kebijakan IAM

Kebijakan adalah entitas AWS yang, ketika dilampirkan ke identitas atau sumber daya, menentukan izin mereka. AWS mengevaluasi kebijakan ini ketika seorang kepala sekolah, seperti pengguna, membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak.

Kebijakan disimpan AWS sebagai dokumen JSON yang dilampirkan ke prinsipal sebagai kebijakan berbasis identitas atau ke sumber daya sebagai kebijakan berbasis sumber daya. Identity-based kebijakan termasuk kebijakan AWS terkelola, kebijakan yang dikelola pelanggan, dan kebijakan sebaris.

Anda dapat membuat dan mengedit kebijakan yang dikelola pelanggan Konsol Manajemen AWS menggunakan opsi editor Visual dan JSON. Saat Anda melihat kebijakan di Konsol Manajemen AWS, Anda dapat melihat ringkasan izin yang diberikan oleh kebijakan tersebut. Anda dapat menggunakan editor visual dan ringkasan kebijakan untuk membantu Anda mendiagnosis dan memperbaiki kesalahan umum yang ditemukan saat mengelola kebijakan IAM.

Harap diingat bahwa semua kebijakan IAM disimpan menggunakan sintaksis yang dimulai dengan aturan JavaScript Object Notation (JSON). Anda tidak perlu memahami sintaksis ini untuk membuat atau mengelola kebijakan Anda. Anda dapat membuat dan mengedit kebijakan menggunakan editor visual di Konsol Manajemen AWS. Untuk mempelajari lebih lanjut tentang sintaksis JSON di kebijakan IAM, lihat Tata bahasa kebijakan IAM JSON.

Pemecahan Masalah Topik Kebijakan IAM

- Atasi masalah menggunakan editor visual
  - Restrukturisasi kebijakan
  - Memilih ARN sumber daya di editor visual
  - Menolak izin di editor visual
  - Menentukan beberapa layanan di editor visual
  - Mengurangi ukuran kebijakan Anda di editor visual
  - Perbaikan layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual
- Memecahkan masalah dengan ringkasan kebijakan
  - Ringkasan kebijakan hilang

  - Ringkasan kebijakan mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal
  - Layanan tidak mendukung ringkasan kebijakan IAM
  - Kebijakan saya tidak memberikan izin yang diharapkan
- Pemecahan masalah manajemen kebijakan
  - Melampirkan atau melepaskan kebijakan di akun IAM
  - Mengubah kebijakan untuk identitas IAM Anda berdasarkan aktivitas
- Memecahkan masalah pada dokumen kebijakan JSON
  - Memvalidasi kebijakan Anda
  - Saya tidak memiliki izin untuk validasi kebijakan di editor JSON
  - Lebih dari satu objek kebijakan JSON
  - Lebih dari satu elemen pernyataan JSON
  - Lebih dari satu efek, tindakan, atau elemen sumber daya dalam elemen pernyataan JSON
  - Elemen versi JSON hilang

Atasi masalah menggunakan editor visual

Saat membuat atau mengedit kebijakan yang dikelola pelanggan, Anda dapat menggunakan informasi di editor Visual untuk membantu memecahkan masalah kesalahan dalam kebijakan Anda. Untuk melihat contoh penggunaan editor visual untuk membuat kebijakan, lihat the section called “Mengontrol akses ke identitas”.

Restrukturisasi kebijakan

Saat Anda membuat kebijakan, AWS memvalidasi, memproses, dan mengubah kebijakan sebelum menyimpannya. Saat kebijakan diambil, AWS ubah kembali ke format yang dapat dibaca manusia tanpa mengubah izin. Hal ini dapat mengakibatkan perbedaan dalam apa yang Anda lihat di editor visual kebijakan atau tab JSON.

- Blok izin editor visual dapat ditambahkan, dihapus, atau disusun ulang, dan konten di dalam blok dapat dioptimalkan.
- Di tab JSON, ruang kosong yang tidak signifikan dapat dihapus, dan elemen di dalam peta JSON dapat disusun ulang. Selain itu, Akun AWS ID dalam elemen utama dapat diganti dengan Nama Sumber Daya Amazon (ARN) dari Pengguna root akun AWS.

Karena kemungkinan perubahan ini, Anda tidak boleh membandingkan dokumen kebijakan JSON sebagai string.

Saat Anda membuat kebijakan yang dikelola pelanggan di Konsol Manajemen AWS, Anda dapat memilih untuk bekerja sepenuhnya di editor JSON. Jika Anda tidak pernah mengubah kebijakan di editor Visual dan memilih Ber ikutnya dari editor JSON, kebijakan tersebut cenderung tidak akan direstrukturisasi. Saat Anda menggunakan editor Visual, IAM mungkin merestrukturisasi kebijakan untuk mengoptimalkan tampilannya. Restrukturisasi ini hanya ada pada sesi pengeditan Anda dan tidak disimpan secara otomatis.

Jika kebijakan Anda distrukturkan ulang di sesi pengeditan, IAM menentukan apakah akan menyimpan restrukturisasi berdasarkan situasi berikut:

Menggunakan opsi editor ini	Jika Anda mengedit kebijakan	Dan kemudian pilih Ber ikutnya dari tab ini	Saat Anda memilih Simpan perubahan
Visual	Diedit	Visual	Kebijakan distruktur ulang
Visual	Diedit	JSON	Kebijakan distruktur ulang
Visual	Tidak diedit	Visual	Kebijakan distruktur ulang
JSON	Diedit	Visual	Kebijakan distruktur ulang
JSON	Diedit	JSON	Struktur kebijakan tidak diubah
JSON	Tidak Diedit	JSON	Struktur kebijakan tidak diubah

IAM mungkin merestrukturisasi kebijakan atau kebijakan rumit yang memiliki blok izin atau pernyataan yang memungkinkan beberapa layanan, jenis sumber daya, atau kunci kondisi.

Memilih ARN sumber daya di editor visual

Saat Anda membuat atau mengedit kebijakan menggunakan editor visual, Anda harus memilih layanan terlebih dahulu, kemudian memilih tindakan dari layanan tersebut. Jika layanan dan tindakan yang Anda pilih mendukung pemilihan sumber daya tertentu, editor visual mencantumkan tipe sumber daya yang didukung. Anda kemudian dapat memilih Tambahkan ARN untuk memberikan detail tentang sumber daya Anda. Anda dapat memilih dari opsi berikut untuk menambahkan ARN untuk tipe sumber daya.

- Gunakan pembuat ARN - Anda mungkin melihat bidang yang berbeda untuk membangun ARN Anda berdasarkan jenis sumber daya. Anda juga dapat memilih Apa pun untuk memberikan izin bagi nilai apa pun untuk pengaturan yang ditentukan. Misalnya, jika Anda memilih grup tingkat akses Baca Amazon EC2, maka tindakan di kebijakan Anda mendukung jenis sumber daya `instance`. Berikan Wilayah, Akun, dan InstanceId nilai untuk sumber daya Anda. Kebijakan ini memberikan izin ke instans apa pun di akun Anda jika Anda memberikan ID akun dan memilih Apa saja untuk ID Wilayah dan instans.
- Ketik atau tempel ARN – Anda dapat menentukan sumber daya berdasarkan Amazon Resource Name (ARN). Anda dapat memasukkan karakter wildcard (*) di bidang ARN mana pun (di antara setiap pasangan titik dua). Untuk informasi selengkapnya, lihat Elemen kebijakan IAM JSON: Sumber daya.

Menolak izin di editor visual

Secara default, kebijakan yang Anda buat menggunakan editor visual memungkinkan tindakan yang Anda pilih. Untuk menolak tindakan yang dipilih, pilih Beralih ke menolak izin. Karena permintaan dit olak secara default, sebaiknya Anda mengizinkan izin hanya untuk tindakan dan sumber daya yang dibutuhkan pengguna. Anda harus membuat pernyataan tolak hanya jika Anda ingin mengganti izin secara terpisah yang diizinkan oleh pernyataan atau kebijakan lain. Kami menyarankan agar Anda membatasi jumlah penolakan izin seminimal mungkin karena dapat meningkatkan kesulitan dalam memecahkan masalah izin. Untuk informasi lebih lanjut tentang cara IAM mengevaluasi logika kebijakan, lihat Logika evaluasi kebijakan.

> **Note**
>
> Secara default, hanya yang Pengguna root akun AWS memiliki akses ke semua sumber daya di akun itu. Jadi, jika Anda belum masuk sebagai pengguna utama, Anda harus memiliki izin yang diberikan oleh kebijakan.

Menentukan beberapa layanan di editor visual

Saat Anda menggunakan editor visual untuk membuat kebijakan, Anda hanya dapat memilih satu layanan pada satu waktu. Ini adalah praktik terbaik karena editor visual kemudian mengizinkankan Anda memilih dari tindakan untuk satu layanan tersebut. Kemudian Anda memilih dari sumber daya yang didukung oleh layanan tersebut dan tindakan yang dipilih. Ini mempermudah pembuatan dan pemecahan masalah kebijakan Anda.

Anda juga dapat menggunakan karakter wildcard (*) untuk menentukan beberapa layanan secara manual. Sebagai contoh, ketik **`Code*`** untuk memberikan izin untuk semua layanan yang dimulai dengan `Code`, seperti `CodeBuild` dan `CodeCommit`. Namun, Anda selanjutnya harus mengetikkan ARN tindakan dan sumber daya untuk melengkapi kebijakan Anda. Sebagai tambahan, saat Anda menyimpan kebijakan Anda, kebijakan itu mungkin distruktur ulang untuk menyertakan setiap layanan di blok izin terpisah.

Atau, untuk menggunakan sintaks JSON (seperti wildcard) untuk layanan, buat, edit, dan simpan kebijakan Anda menggunakan opsi editor JSON.

Mengurangi ukuran kebijakan Anda di editor visual

Saat Anda menggunakan editor visual untuk membuat kebijakan, IAM membuat dokumen JSON untuk menyimpan kebijakan Anda. Anda dapat melihat dokumen ini dengan beralih ke opsi editor JSON. Jika dokumen JSON ini melebihi batas ukuran kebijakan, editor visual menampilkan pesan kesalahan. Anda tidak akan dapat meninjau dan menyimpan kebijakan. Untuk melihat batasan IAM mengenai ukuran kebijakan yang dikelola, lihat Batas karakter IAM dan STS.

Untuk mengurangi ukuran kebijakan Anda di editor visual, edit kebijakan Anda atau pindahkan blok izin ke kebijakan lain. Pesan galat menyertakan jumlah karakter yang berisi dokumen kebijakan Anda. Anda dapat menggunakan informasi ini untuk membantu mengurangi ukuran kebijakan Anda.

Perbaikan layanan, tindakan, atau tipe sumber daya yang tidak dikenal di editor visual

Anda mungkin melihat peringatan di editor visual bahwa kebijakan Anda menyertakan jenis layanan, tindakan, atau sumber daya yang tidak dikenali.

> **Note**
>
> IAM meninjau nama layanan, tindakan, dan jenis sumber daya untuk layanan yang mendukung ringkasan kebijakan. Akan tetapi, ringkasan kebijakan Anda mungkin mencakup

nilai sumber daya atau kondisi yang tidak ada. Selalu uji kebijakan Anda dengan simulator kebijakan.

Jika kebijakan Anda mencakup layanan, tindakan, atau jenis sumber daya yang tidak dikenal, salah satu kesalahan berikut telah terjadi:

- Layanan pratinjau – Layanan yang ada di pratinjau tidak mendukung editor visual. Jika Anda berpartisipasi dalam pratinjau, Anda harus mengetikkan tindakan dan ARN sumber daya secara manual untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempelkan dokumen kebijakan JSON.
- Layanan khusus – Layanan khusus tidak mendukung editor visual. Jika Anda menggunakan layanan khusus, Anda harus mengetikkan tindakan dan ARN sumber daya secara manual untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempelkan dokumen kebijakan JSON.
- Layanan tidak mendukung editor visual — Jika kebijakan Anda menyertakan layanan yang tersedia secara umum (GA) yang tidak mendukung editor visual, Anda harus mengetikkan tindakan dan ARN sumber daya secara manual untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempelkan dokumen kebijakan JSON.

  Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan pratinjau atau layanan khusus. Jika layanan yang tidak dikenal tersedia secara umum dan nama dieja dengan benar, maka layanan tersebut tidak mendukung editor visual. Untuk mempelajari cara meminta dukungan editor visual atau ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan kebijakan IAM.
- Tindakan tidak mendukung editor visual — Jika kebijakan Anda menyertakan layanan yang didukung dengan tindakan yang tidak didukung, Anda harus mengetikkan tindakan dan ARN sumber daya secara manual untuk menyelesaikan kebijakan Anda. Anda dapat mengabaikan peringatan apa pun dan melanjutkan. Atau, Anda dapat memilih opsi editor JSON untuk mengetik atau menempelkan dokumen kebijakan JSON.

  Jika kebijakan Anda menyertakan layanan yang didukung dengan tindakan yang tidak didukung, maka layanan tersebut tidak sepenuhnya mendukung editor visual. Untuk mempelajari cara

meminta dukungan editor visual atau ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan kebijakan IAM.

- Jenis sumber daya tidak mendukung editor visual – Jika kebijakan Anda mencakup tindakan yang didukung dengan jenis sumber daya yang tidak didukung, Anda dapat mengabaikan peringatan dan melanjutkan. Namun, IAM tidak dapat mengonfirmasi bahwa Anda telah menyertakan sumber daya untuk semua tindakan yang Anda pilih, dan Anda mungkin melihat peringatan tambahan.
- Salah ketik – Saat Anda mengetikkan layanan, tindakan, atau sumber daya secara manual dalam editor visual, Anda dapat membuat kebijakan yang mencakup kesalahan ketik. Kami menyarankan Anda menggunakan editor visual dengan memilih dari daftar layanan dan tindakan. Kemudian, lengkapi bagian sumber daya sesuai dengan petunjuknya. Jika layanan tidak sepenuhnya mendukung editor visual, Anda mungkin harus mengetikkan bagian kebijakan secara manual.

  Jika Anda yakin bahwa kebijakan Anda tidak berisi satu pun kesalahan di atas, maka kebijakan Anda mungkin mengandung kesalahan ketik. Periksa masalah berikut:

  - Nama jenis layanan, tindakan, dan sumber daya salah eja, seperti al `s2` ih- `ListMyBuckets` alih `s3` atau bukan `ListAllMyBuckets`
  - Teks yang tidak perlu di ARN, seperti `arn:aws:s3: : :*`
  - Titik dua hilang dalam tindakan, seperti `iam.CreateUser`

  Anda dapat mengevaluasi kebijakan yang mungkin menyertakan kesalahan ketik dengan memilih Ber ikutnya untuk meninjau ringkasan kebijakan. Kemudian, konfirmasikan apakah kebijakan tersebut memberikan izin yang Anda inginkan.

Memecahkan masalah dengan ringkasan kebijakan

Anda dapat mendiagnosis dan menyelesaikan masalah terkait rangkuman kebijakan.

Ringkasan kebijakan hilang

Konsol IAM mencakup tabel ringkasan kebijakan yang menjelaskan tingkat akses, sumber daya, dan kondisi yang diperbolehkan atau ditolak untuk setiap layanan dalam kebijakan. Kebijakan dirangkum dalam tiga tabel: ringkasan kebijakan, ringkasan layanan, dan ringkasan tindakan.

Tabel ringkasan kebijakan mencakup daftar layanan dan ringkasan izin yang ditentukan oleh kebijakan yang dipilih. Ringkasan kebijakan tersedia di halaman detail kebijakan untuk kebijakan terlampir, atau di halaman Kebijakan untuk kebijakan terkelola. Jika AWS tidak dapat merender

ringkasan untuk kebijakan, konsol akan menampilkan dokumen kebijakan JSON dan kesalahan berikut:

Ringkasan untuk kebijakan ini tidak dapat dibuat. Anda masih dapat melihat atau mengedit dokumen kebijakan JSON.

Jika kebijakan Anda tidak mencakup ringkasan, salah satu kesalahan berikut telah terjadi:

- Elemen kebijakan yang tidak didukung – IAM tidak mendukung pembuatan ringkasan kebijakan untuk kebijakan yang mencakup salah satu dari elemen kebijakan berikut ini:
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Tidak ada izin kebijakan – Jika kebijakan tidak menyediakan izin yang efektif, maka ringkasan kebijakan tidak dapat dibuat. Misalnya, jika kebijakan mencakup satu pernyataan dengan elemen `"NotAction": "*"`, maka akses ke semua tindakan akan diberikan kecuali “semua tindakan” (*). Ini berarti tidak ada akses `Deny` atau `Allow` yang diberikan.

  > **Note**
  >
  > Hati-hati saat menggunakan elemen kebijakan ini seperti`NotPrincipal`,`NotAction`, dan`NotResource`. Untuk informasi selengkapnya tentang elemen-elemen kebijakan, lihat Referensi elemen kebijakan IAM JSON.

  Jika Anda menyediakan layanan dan sumber daya yang tidak cocok, Anda dapat membuat kebijakan yang tidak memberikan izin efektif. Hal ini dapat terjadi ketika Anda menentukan tindakan dalam satu layanan dan sumber daya dari layanan lain. Dalam hal ini, ringkasan kebijakan muncul. Satu-satunya indikasi bahwa terdapat masalah adalah kolom sumber daya di ringkasan dapat mencakup sumber daya dari layanan yang berbeda. Jika kolom ini memuat sumber daya yang tidak sesuai, maka Anda harus meninjau kebijakan Anda untuk kesalahan. Uji kebijakan Anda dengan simulator kebijakan untuk lebih memahami kebijakan.

Ringkasan kebijakan mencakup layanan, tindakan, atau tipe sumber daya yang tidak dikenal

Di konsol IAM, jika ringkasan kebijakan menyer takan simbol peringatan (), kebijakan tersebut mungkin menyertakan jenis layanan, tindakan, atau sumber daya yang tidak dikenali. Untuk mempelajari tentang peringatan di dalam ringkasan kebijakan, lihat Ringkasan kebijakan (daftar layanan).

> **Note**
>
> IAM meninjau nama layanan, tindakan, dan jenis sumber daya untuk layanan yang mendukung ringkasan kebijakan. Akan tetapi, ringkasan kebijakan Anda mungkin mencakup nilai sumber daya atau kondisi yang tidak ada. Selalu uji kebijakan Anda dengan simulator kebijakan.

Jika kebijakan Anda mencakup layanan, tindakan, atau jenis sumber daya yang tidak dikenal, salah satu kesalahan berikut telah terjadi:

- Layanan pratinjau – Layanan yang ada di pratinjau tidak mendukung rangkuman kebijakan.
- Layanan khusus – Layanan khusus tidak mendukung rangkuman kebijakan.
- Layanan tidak mendukung rangkuman – Jika kebijakan Anda mencakup layanan yang tersedia secara umum (GA) yang tidak mendukung rangkuman kebijakan, maka layanan tersebut termasuk dalam bagian Layanan yang tidak dikenal di tabel ringkasan kebijakan. Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan pratinjau atau layanan khusus. Jika layanan yang tidak dikenal tersedia secara umum dan nama dieja dengan benar, maka layanan ini tidak mendukung ringkasan kebijakan IAM. Untuk mempelajari cara meminta dukungan ringkasan kebijakan untuk layanan GA, lihat Layanan tidak mendukung ringkasan kebijakan IAM.
- Tindakan tidak mendukung rangkuman – Jika kebijakan Anda mencakup layanan yang didukung dengan tindakan yang tidak didukung, maka tindakan tersebut termasuk dalam bagian Tindakan yang tidak dikenal pada tabel ringkasan layanan. Untuk mempelajari tentang peringatan dalam rangkuman layanan, lihat Ringkasan layanan (daftar tindakan).
- Jenis sumber daya tidak mendukung rangkuman – Jika kebijakan Anda mencakup tindakan yang didukung dengan jenis sumber daya yang tidak didukung, maka sumber daya tersebut disertakan dalam bagian Jenis sumber daya yang tidak dikenal pada tabel ringkasan layanan.

Untuk mempelajari tentang peringatan di dalam rangkuman layanan, lihat Ringkasan layanan (daftar tindakan).

- Salah ke tik — AWS memeriksa apakah JSON benar secara sintaksis, dan kebijakan tidak menyertakan kesalahan ketik atau kesalahan lainnya sebagai bagian dari validasi kebijakan. ???

> **Note**
>
> Sebagai praktik terbaik, kami menyarankan Anda menggunakan Penganalisis Akses IAM untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan fungsional. Kami menyarankan Anda membuka kebijakan yang ada dan meninjau serta menyelesaikan rekomendasi validasi kebijakan apa pun.

Layanan tidak mendukung ringkasan kebijakan IAM

Ringkasan kebijakan IAM atau editor visual mungkin tidak mendukung layanan atau tindakan yang tersedia secara umum (GA). Layanan yang tersedia secara umum adalah layanan yang dirilis secara publik dan bukan layanan yang dipratinjau atau khusus. Jika layanan yang tidak dikenal tersedia secara umum dan nama dieja dengan benar, maka layanan tidak mendukung fitur-fitur ini. Jika kebijakan Anda mencakup layanan yang didukung dengan tindakan yang tidak didukung, maka layanan tersebut tidak sepenuhnya mendukung rangkuman kebijakan IAM.

Untuk meminta agar layanan menambahkan ringkasan kebijakan IAM atau dukungan editor visual

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Temukan kebijakan yang mencakup layanan yang tidak didukung:
   - Jika kebijakan tersebut adalah kebijakan terkelola, pilih Kebijakan di panel navigasi. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
   - Jika kebijakan tersebut adalah kebijakan inline yang terlampir pada pengguna, pilih Pengguna di panel navigasi. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat. Dalam tabel kebijakan untuk pengguna, perluas kop untuk ringkasan kebijakan yang ingin Anda lihat.
3. Di sisi kiri pada Konsol Manajemen AWS footer, pilih Umpan Bal ik. Di kotak U mpan Balik untuk IAM, ketik**`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Jika Anda menginginkan lebih dari

satu layanan untuk mendukung ringkasan, ketik **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Untuk meminta agar layanan menambahkan dukungan ringkasan kebijakan IAM untuk tindakan yang hilang

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Temukan kebijakan yang mencakup layanan yang tidak didukung:
   - Jika kebijakan tersebut adalah kebijakan terkelola, pilih Kebijakan di panel navigasi. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda lihat.
   - Jika kebijakan tersebut adalah kebijakan inline yang terlampir pada pengguna, pilih Pengguna di panel navigasi. Dalam daftar pengguna, pilih nama pengguna yang kebijakannya ingin Anda lihat. Dalam tabel kebijakan untuk pengguna, pilih nama kebijakan yang ingin Anda lihat untuk memperluas ringkasan kebijakan.
3. Dalam ringkasan kebijakan, pilih nama layanan yang menyertakan tindakan yang tidak didukung.
4. Di sisi kiri pada Konsol Manajemen AWS footer, pilih Umpan Balik. Di kotak Umpan Balik untuk IAM, ketik**`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Jika Anda ingin melaporkan lebih dari satu tindakan yang tidak didukung, ketik **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Untuk meminta agar layanan lain mencakup tindakan yang hilang, ulangi tiga langkah terakhir.

Kebijakan saya tidak memberikan izin yang diharapkan

Untuk menetapkan izin bagi pengguna, grup, peran, atau sumber daya, Anda membuat kebijakan, yaitu dokumen yang mendefinisikan izin. Dokumen kebijakan mencakup elemen-elemen berikut ini:

- Efek – apakah kebijakan mengizinkan atau menolak akses
- Tindakan – daftar tindakan yang diperbolehkan atau ditolak oleh kebijakan tersebut
- Sumber Daya – daftar sumber daya yang memungkinkan terjadinya tindakan
- Kondisi (Opsional) – situasi yang mengatur pemberian izin oleh kebijakan

Untuk mempelajari tentang elemen kebijakan ini dan elemen kebijakan lainnya, lihat Referensi elemen kebijakan IAM JSON.

Untuk memberikan akses, kebijakan Anda harus menentukan tindakan dengan sumber daya yang didukung. Jika kebijakan Anda juga mencakup suatu kondisi, kondisi tersebut harus mencakup kunci kondisi global atau harus berlaku untuk tindakan tersebut. Untuk mempelajari sumber daya mana yang didukung oleh tindakan, lihat Dokumentasi AWS untuk layanan Anda. Untuk mempelajari kondisi mana yang didukung oleh tindakan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Periksa apakah kebijakan Anda mendefinisikan tindakan, sumber daya, atau kondisi yang tidak memberikan izin. Lihat ringkasan kebijakan untuk kebijakan Anda menggunakan konsol IAM di https://console.aws.amazon.com/iam/. Anda dapat menggunakan ringkasan kebijakan untuk mengidentifikasi dan memperbaiki masalah di kebijakan Anda.

Ada beberapa alasan mengapa elemen mungkin tidak memberikan izin meskipun ditetapkan dalam kebijakan IAM:

- Tindakan didefinisikan tanpa sumber daya yang berlaku
- Sumber daya didefinisikan tanpa tindakan yang berlaku
- Suatu kondisi didefinisikan tanpa tindakan yang berlaku

Untuk melihat contoh ringkasan kebijakan yang mencakup peringatan, lihat the section called "Ringkasan kebijakan (daftar layanan)".

Tindakan didefinisikan tanpa sumber daya yang sesuai

Kebijakan di bawah ini menjelaskan semua tindakan `ec2:Describe*` dan menentukan sumber daya spesifik. Tidak ada `ec2:Describe` tindakan yang diberikan karena tidak ada tindakan yang mendukung izin tingkat sumber daya. Resource-level izin berarti tindakan mendukung sumber daya menggunakan ARN di `Resource` elemen kebijakan. Jika tindakan tidak mendukung izin tingkat sumber daya, maka pernyataan dalam kebijakan tersebut harus menggunakan wildcard (`*`) di elemen `Resource`. Untuk mempelajari layanan mana yang mendukung izin tingkat sumber daya, lihat AWS layanan yang bekerja dengan IAM.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "arn:aws:ec2:us-east-2:111122223333:instance/*"
        }
    ]
}
```

Kebijakan ini tidak memberikan izin apa pun, dan ringkasan kebijakan mencakup kesalahan berikut ini:

`This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Untuk memperbaiki kebijakan ini, Anda harus menggunakan `*` di elemen `Resource`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
}
```

Sumber daya ditetapkan tanpa tindakan yang sesuai

Kebijakan di bawah ini mendefinisikan sumber daya bucket Amazon S3 tetapi tidak mencakup tindakan S3 yang dapat dilakukan pada sumber daya tersebut. Kebijakan ini juga memberikan akses penuh ke semua CloudFront tindakan Amazon.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
        }
    ]
}
```

Kebijakan ini memberikan izin untuk semua CloudFront tindakan. Tetapi karena kebijakan mendefinisikan sumber daya `amzn-s3-demo-bucket` S3 tanpa mendefinisikan tindakan S3 apa pun, ringkasan kebijakan mencakup peringatan berikut:

`This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Untuk memperbaiki kebijakan ini guna memberikan izin bucket S3, Anda harus menetapkan tindakan S3 yang dapat dilakukan pada sumber daya bucket.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudfront:*",
                "s3:CreateBucket",
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:cloudfront:*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
```

```
        }
    ]
}
```

Sebagai alternatif, untuk memperbaiki kebijakan ini agar hanya memberikan CloudFront izin, hapus sumber daya S3.

Kondisi didefinisikan tanpa tindakan yang berlaku

Kebijakan di bawah ini mendefinisikan dua tindakan Amazon S3 untuk semua sumber daya S3, jika awalan S3 sama `custom` dan ID versi sama`1234`. Namun, kunci kondisi `s3:VersionId` digunakan untuk penandaan versi objek dan tidak didukung oleh tindakan bucket didefinisikan. Untuk mempelajari kondisi mana yang didukung oleh tindakan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan untuk melihat dokumentasi layanan untuk kunci kondisi.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
```

```
}
```

Kebijakan ini memberikan izin untuk tindakan `s3:ListBucketVersions` dan tindakan `s3:ListBucket` jika nama bucket mencakup prefiks `custom`. Tetapi karena kondisi `s3:VersionId` tidak didukung oleh salah satu tindakan yang ditetapkan, ringkasan kebijakan mencakup kesalahan berikut:

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

Untuk memperbaiki kebijakan ini untuk menggunakan penandaan versi objek S3, Anda harus menentukan tindakan S3 yang mendukung kunci kondisi `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Kebijakan ini memberikan izin untuk setiap tindakan dan kondisi dalam kebijakan tersebut. Namun, kebijakan ini masih tidak memberikan izin apa pun karena tidak ada kasus ketika tindakan tunggal sesuai dengan kedua kondisi. Sebaliknya, Anda harus membuat dua pernyataan terpisah yang masing-masing hanya menyertakan tindakan kondisi yang sesuai dengannya.

Untuk memperbaiki kebijakan ini, buat dua pernyataan. Pernyataan pertama mencakup tindakan yang mendukung kondisi `s3:prefix`, dan pernyataan kedua mencakup tindakan yang mendukung kondisi `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Pemecahan masalah manajemen kebijakan

Anda dapat mendiagnosis dan menyelesaikan masalah terkait manajemen kebijakan.

Melampirkan atau melepaskan kebijakan di akun IAM

Beberapa AWS kebijakan terkelola terkait dengan layanan. Kebijakan ini hanya digunakan dengan peran yang berkaitan dengan layanan untuk layanan itu. Di konsol IAM, saat Anda melihat halaman detail kebijakan, halaman tersebut menyertakan spanduk untuk menunjukkan bahwa kebijakan tersebut ditautkan ke layanan. Anda tidak dapat melampirkan kebijakan ini ke pengguna, grup, atau peran di dalam IAM. Saat Anda membuat peran terkait layanan, kebijakan ini secara otomatis dilampirkan ke peran baru Anda. Karena kebijakan tersebut diperlukan, Anda tidak dapat memisahkan kebijakan dari peran yang terkait dengan layanan.

Mengubah kebijakan untuk identitas IAM Anda berdasarkan aktivitas

Anda dapat memperbarui kebijakan untuk identitas IAM Anda (pengguna, grup, dan peran) berdasarkan aktivitasnya. Untuk melakukan ini, lihat peristiwa akun Anda di Ri CloudTrail wayat acara. CloudTrail log peristiwa menyertakan informasi peristiwa terperinci yang dapat Anda gunakan untuk mengubah izin kebijakan.

Pengguna atau peran sedang mencoba melakukan tindakan di AWS dan permintaan itu ditolak.

- Pertimbangkan apakah pengguna atau peran harus memiliki izin untuk melakukan tindakan. Jika demikian, Anda dapat menambahkan tindakan dan bahkan ARN dari sumber daya yang mereka coba akses ke kebijakan mereka.

Pengguna atau peran memiliki izin yang tidak mereka gunakan.

- Pertimbangkan untuk menghapus izin tersebut dari kebijakan mereka. Pastikan bahwa kebijakan Anda memberikan hak istimewa terkecil yang diperlukan untuk hanya melakukan tindakan yang diperlukan.

Untuk informasi selengkapnya tentang penggunaan CloudTrail, lihat Meli CloudTrail hat Acara di CloudTrail Konsol di Panduan AWS CloudTrail Pengguna.

Memecahkan masalah pada dokumen kebijakan JSON

Anda dapat mendiagnosis dan menyelesaikan masalah terkait dokumen kebijakan JSON.

Memvalidasi kebijakan Anda

Saat Anda membuat atau mengedit kebijakan JSON, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari lebih lanjut tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Saya tidak memiliki izin untuk validasi kebijakan di editor JSON

Di dalam Konsol Manajemen AWS, Anda mungkin menerima kesalahan berikut jika Anda tidak memiliki izin untuk melihat hasil validasi kebijakan Penganalisis Akses IAM:

```
You need permissions. You do not have the permissions required to perform
this operation. Ask your administrator to add permissions.
```

Untuk memperbaiki kesalahan ini, minta administrator Anda untuk menambahkan izin `access-analyzer:ValidatePolicy` untuk Anda.

Lebih dari satu objek kebijakan JSON

Kebijakan IAM harus terdiri dari hanya satu objek JSON. Anda menunjukkan sebuah objek dengan menempatkan rungkup { } di sekitarnya. Anda dapat membuat sarang objek lain dalam objek JSON dengan menyematkan kurung {} tambahan di dalam pasangan luar. Kebijakan hanya harus berisi satu pasang kurung kurung {} terluar. Contoh berikut salah karena berisi dua objek di tingkat atas (dipanggil masuk*red*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
```

```
}
{
  "Statement": {
     "Effect": "Allow",
     "Action": "s3:*",
     "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

Namun, Anda dapat memenuhi maksud dari contoh sebelumnya dengan menggunakan tata bahasa kebijakan yang benar. Alih-alih memasukkan dua objek kebijakan lengkap, masing-masing dengan elemen `Statement`, Anda dapat menggabungkan kedua blok menjadi satu elemen `Statement`. Elemen `Statement` memiliki susunan dua objek sebagai nilainya, seperti ditunjukkan dalam contoh berikut (disebut dalam huruf tebal):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
      ]
    }
```

Lebih dari satu elemen pernyataan JSON

Kesalahan ini pada awalnya mungkin tampak seperti variasi pada bagian sebelumnya. Namun, secara sintaksis ini adalah tipe kesalahan yang berbeda. Contoh berikut hanya memiliki satu objek kebijakan yang ditandai dengan sepasang rungkup { } di tingkat atas. Namun, objek tersebut berisi dua elemen `Statement` di dalamnya.

Kebijakan IAM harus berisi hanya satu elemen `Statement`, yang terdiri atas nama (`Statement`) yang muncul di sebelah kiri titik dua, diikuti dengan nilainya di sebelah kanan. Nilai dari elemen `Statement` harus berupa objek, yang ditandai dengan rungkup { }, yang berisi satu elemen `Effect`, satu elemen `Action`, dan satu elemen `Resource`. Contoh berikut tidak benar karena berisi dua `Statement` elemen dalam objek kebijakan (dipanggil masuk*red*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Objek nilai dapat berupa himpunan berbagai objek nilai. Untuk memecahkan masalah ini, gabungkan kedua elemen `Statement` ke dalam satu elemen dengan himpunan objek, seperti ditunjukkan dalam contoh berikut (disebut dalam huruf tebal):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
```

```
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
     ]
    }
```

Nilai dari elemen `Statement` merupakan himpunan objek. Rangkaian dalam contoh ini terdiri dari dua objek, yang masing-masing adalah nilai yang benar untuk elemen `Statement`. Setiap objek di himpunan dipisahkan dengan koma.

Lebih dari satu efek, tindakan, atau elemen sumber daya dalam elemen pernyataan JSON

Pada sisi nilai `Statement` name/value pasangan, objek harus terdiri dari hanya satu `Effect` elemen, satu `Action` elemen, dan satu `Resource` elemen. Kebijakan berikut ini salah karena memiliki dua `Effect` elemen dalam`Statement`:

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> Mesin kebijakan tidak mengizinkan kesalahan tersebut di kebijakan baru atau yang telah diedit. Namun demikian, mesin kebijakan terus mengizinkan kebijakan yang disimpan sebelum mesin diperbarui. Perilaku kebijakan yang ada dengan kesalahan tersebut adalah sebagai berikut:
>
> - Beberapa elemen `Effect`: hanya elemen `Effect` terakhir yang diamati. Yang lainnya diabaikan.

- Beberapa `Action` elemen: semua `Action` elemen digabungkan secara internal dan diperlakukan seolah-olah mereka adalah daftar tunggal.
- Beberapa `Resource` elemen: semua `Resource` elemen digabungkan secara internal dan diperlakukan seolah-olah mereka adalah daftar tunggal.

Mesin kebijakan tidak memungkinkan Anda untuk menyimpan kebijakan dengan kesalahan sintaksis. Perbaiki kesalahan dalam kebijakan sebelum menyimpan. Tinjau dan perbaiki setiap rekomendasi validasi kebijakan untuk kebijakan Anda.

Dalam setiap kasus, solusinya adalah menghapus elemen ekstra yang salah. Untuk elemen `Effect`, ini mudah: jika Anda ingin contoh sebelumnya untuk menolak izin bagi instans Amazon EC2, maka Anda harus menghapus baris `"Effect": "Allow",` dari kebijakan, sebagai berikut:

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:*",
        "Resource": "*"
      }
    }
```

Namun, jika elemen duplikat adalah `Action` atau `Resource`, maka resolusinya dapat lebih rumit. Anda mungkin memiliki beberapa tindakan yang ingin Anda izinkan (atau tolak) izinnya, atau Anda mungkin ingin mengontrol akses ke beberapa sumber daya. Misalnya, contoh berikut salah karena memiliki beberapa `Resource` elemen (dipanggil masuk*red*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
```

```
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Masing-masing elemen wajib di objek nilai elemen `Statement` hanya bisa muncul sekali. Solusinya adalah menempatkan setiap nilai dalam himpunan. Contoh berikut menggambarkan hal ini dengan membuat dua elemen sumber daya terpisah menjadi satu elemen `Resource` dengan susunan sebagai objek nilai (disebut dalam huruf tebal):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      }
    }
```

Elemen versi JSON hilang

Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Sebagai perbandingan, versi kebijakan dibuat saat Anda mengubah kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola. Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat Elemen kebijakan IAM JSON: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Peningkatan versi IAM”.

Seiring berkembangnya AWS fitur, kemampuan baru ditambahkan ke kebijakan IAM untuk mendukung fitur-fitur tersebut. Terkadang, pembaruan sintaksis kebijakan mencakup nomor versi baru. Jika Anda menggunakan fitur tata bahasa kebijakan yang lebih baru dalam kebijakan Anda, maka Anda harus memberi tahu mesin pembatas kebijakan versi mana yang Anda gunakan.

Versi kebijakan default adalah “17-10-2008.” Jika Anda ingin menggunakan fitur kebijakan yang diperkenalkan setelahnya, Anda harus menentukan nomor versi yang mendukung fitur yang Anda inginkan. Sebaiknya Anda selalu menyertakan nomor versi sintaksis kebijakan terbaru, yang saat ini adalah `"Version": "2012-10-17"`.

Misalnya, kebijakan berikut salah karena menggunakan variabel kebijakan `${...}` di ARN untuk sumber daya. Tetapi gagal menentukan versi sintaks kebijakan yang mendukung variabel kebijakan (dipanggil masuk*red*):

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Menambahkan elemen `Version` di atas kebijakan dengan nilai `2012-10-17`, API IAM versi pertama yang mendukung variabel kebijakan, menyelesaikan masalah ini (disebut dalam huruf tebal):

JSON

```
{
  "Version":"2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Memecahkan Masalah Kunci Sandi dan Kunci Keamanan FIDO

Gunakan informasi di sini untuk membantu Anda mendiagnosis masalah umum yang mungkin Anda temui saat bekerja dengan kunci keamanan FIDO2.

Topik

- Saya tidak dapat mengaktifkan kunci keamanan FIDO saya
- Saya tidak bisa masuk menggunakan kunci keamanan FIDO saya
- Saya kehilangan atau merusak kunci keamanan FIDO saya
- Masalah lainnya

Saya tidak dapat mengaktifkan kunci keamanan FIDO saya

Pelajari solusi berikut tergantung pada status Anda sebagai pengguna IAM atau administrator sistem

Pengguna IAM:

Jika Anda tidak dapat mengaktifkan kunci keamanan FIDO, periksa hal berikut:

- Apakah Anda menggunakan konfigurasi yang didukung?

  IAM mendukung perangkat keamanan FIDO2 yang terhubung ke perangkat Anda melalui USB,Bluetooth, atau NFC. IAM juga mendukung otentikator platform seperti TouchID atau FaceID. IAM tidak mendukung pendaftaran kunci sandi lokal untuk Windows Hello. Untuk membuat dan menggunakan kunci sandi, pengguna Windows harus menggunakan otentikasi lintas perangkat di mana Anda menggunakan kunci sandi dari satu perangkat seperti perangkat seluler atau kunci keamanan perangkat keras untuk masuk di perangkat lain seperti laptop.

  Untuk informasi tentang perangkat dan browser yang dapat Anda gunakan dengan WebAuthn dan AWS, lihatKonfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan.
- Apakah Anda menggunakan plugin peramban?

  AWS tidak mendukung penggunaan plugin untuk menambahkan dukungan WebAuthn browser. Sebagai gantinya, gunakan browser yang menawarkan dukungan asli WebAuthn standar.

  Bahkan jika Anda menggunakan browser yang didukung, Anda mungkin memiliki plugin yang tidak kompatibel dengannya WebAuthn. Plugin yang tidak kompatibel dapat mencegah Anda mengaktifkan dan menggunakan kunci FIDO-compliant keamanan Anda. Nonaktifkan plugin yang mungkin tidak kompatibel dan mulai ulang browser Anda. Kemudian, coba lagi mengaktifkan kunci keamanan FIDO.
- Apakah Anda memiliki izin yang sesuai?

  Jika tidak ada satu pun masalah kompatibilitas di atas, Anda mungkin tidak memiliki izin yang sesuai. Hubungi administrator sistem Anda.

Administrator sistem

Jika pengguna IAM Anda tidak dapat mengaktifkan kunci keamanan FIDO mereka meskipun menggunakan konfigurasi yang didukung, periksa izin mereka. Untuk contoh terperinci, lihat Tutorial IAM: Izinkan pengguna untuk mengelola kredensi dan pengaturan MFA mereka.

Saya tidak bisa masuk menggunakan kunci keamanan FIDO saya

Jika Anda tidak dapat masuk Konsol Manajemen AWS menggunakan kunci keamanan FIDO Anda, lihat Konfigurasi yang didukung untuk menggunakan kunci sandi dan kunci keamanan terlebih dahulu. Jika Anda menggunakan konfigurasi yang didukung tetapi tidak dapat masuk, hubungi administrator sistem untuk mendapatkan bantuan.

Saya kehilangan atau merusak kunci keamanan FIDO saya

Hingga delapan perangkat MFA dari kombinasi jenis MFA yang saat ini didukung dapat ditetapkan ke pengguna. Dengan beberapa perangkat MFA, Anda hanya perlu satu perangkat MFA untuk masuk ke. Konsol Manajemen AWS Mengganti kunci keamanan FIDO mirip dengan mengganti token TOTP perangkat keras. Jika Anda kehilangan atau merusak perangkat MFA apa pun, lihatMemulihkan identitas yang dilindungi MFA di IAM.

Masalah lainnya

Jika Anda memiliki masalah dengan kunci keamanan FIDO yang tidak tercakup di sini, lakukan salah satu hal berikut:

- Pengguna IAM: Hubungi administrator sistem Anda.
- Akun AWS pengguna root: Hubungi AWS Dukungan.

Memecahkan masalah peran IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan peran IAM.

Topik

- Saya tidak dapat mengsumsikan peran
- Peran baru muncul di AWS akun
- Saya tidak dapat mengedit atau menghapus peran di Akun AWS

- Saya tidak berwenang untuk melakukan: iam: PassRole
- Mengapa saya tidak bisa mengambil peran dengan sesi 12 jam? (AWS CLI, AWS API)
- Saya menerima kesalahan saat mencoba beralih peran di konsol IAM
- Peran saya memiliki kebijakan yang memungkinkan saya melakukan tindakan, tetapi saya mendapatkan “akses ditolak”
- Layanan tidak membuat versi kebijakan default peran tersebut
- Tidak ada kasus penggunaan untuk peran layanan di konsol

Saya tidak dapat mengsumsikan peran

Periksa hal berikut:

- Untuk mengizinkan pengguna mengambil peran saat ini lagi dalam sesi peran, tentukan peran ARN atau Akun AWS ARN sebagai prinsipal dalam kebijakan kepercayaan peran. Layanan AWS yang menyediakan sumber daya komputasi seperti Amazon EC2, Amazon ECS, Amazon EKS, dan Lambda memberikan kredenSIAL sementara dan secara otomatis memperbarui kredenSIAL ini. Ini memastikan bahwa Anda selalu memiliki kumpulan kredenSIAL yang valid. Untuk layanan ini, tidak perlu mengambil peran saat ini lagi untuk mendapatkan kredenSIAL sementara. Namun, jika Anda ingin mener uskan tag sesi atau kebijakan sesi, Anda harus mengambil peran saat ini lagi. Untuk mempelajari cara memodifikasi kebijakan kepercayaan peran untuk menambahkan peran utama ARN atau Akun AWS ARN, lihatMemperbarui kebijakan kepercayaan peran.
- Ketika Anda mengambil peran menggunakan Konsol Manajemen AWS, pastikan untuk menggunakan nama yang tepat dari peran Anda. Nama peran peka huruf besar/kecil.
- Ketika Anda mengambil peran menggunakan AWS STS API atau AWS CLI, pastikan untuk menggunakan nama persis peran Anda di ARN. Nama peran peka huruf besar/kecil.
- Saat Anda mengambil peran menggunakan penyedia identitas SAML-based federasi dan enkripsi SAML diaktifkan, pastikan Anda telah mengunggah kunci dekripsi pribadi yang valid untuk penyedia identitas SAML. Untuk informasi selengkapnya, lihat Kelola kunci enkripsi SAML.
- Verifikasi bahwa kebijakan IAM Anda mengizinkan memanggil `sts:AssumeRole` untuk peran yang ingin Anda asumsikan. Elemen `Action` kebijakan IAM Anda harus mengizinkan Anda memanggil tindakan `AssumeRole`. Selain itu, elemen `Resource` kebijakan IAM Anda harus menentukan peran yang ingin Anda asumsikan. Misalnya, elemen `Resource` dapat menentukan peran berdasarkan Amazon Resource Name (ARN) atau berdasarkan wildcard (*). Misalnya, setidaknya satu kebijakan yang berlaku bagi Anda harus memberikan izin yang serupa dengan yang berikut ini:

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Verifikasikan bahwa identitas IAM Anda ditandai dengan tag apa pun yang diwajibkan oleh kebijakan IAM. Misalnya, dalam izin kebijakan berikut, elemen `Condition` mewajibkan Anda, sebagai prinsipal yang meminta untuk mengasumsikan peran tersebut, memiliki tanda khusus. Anda harus ditandai dengan `department = HR` atau `department = CS`. Jika tidak, Anda tidak dapat mengambil peran tersebut. Untuk mempelajari tentang menandai pengguna dan peran IAM, lihat the section called “Tag untuk sumber daya IAM”.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
            "HR",
            "CS"
        ]}}
```

- Verifikasi bahwa Anda memenuhi semua kondisi yang ditentukan di kebijakan kepercayaan peran tersebut. `Condition` dapat menentukan tanggal kedaluwarsa, ID eksternal, atau bahwa permintaan harus berasal hanya dari alamat IP tertentu. Pertimbangkan contoh berikut: Jika tanggal saat ini adalah kapan saja setelah tanggal yang ditentukan, kebijakan tidak pernah cocok dan tidak dapat memberi Anda izin untuk mengambil peran tersebut.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- Verifikasi bahwa Akun AWS dari mana Anda memanggil `AssumeRole` adalah entitas tepercaya untuk peran yang Anda ambil. Entitas tepercaya didefinisikan sebagai `Principal` di kebijakan kepercayaan peran. Contoh berikut adalah kebijakan kepercayaan yang terlampir pada peran yang ingin Anda asumsikan. Dalam contoh ini, ID akun dengan pengguna IAM yang Anda gunakan untuk masuk harus 123456789012. Jika nomor akun Anda tidak tercantum di elemen `Principal`

kebijakan kepercayaan peran, maka Anda tidak dapat mengasumsikan peran tersebut. Izin apa pun yang diberikan kepada Anda dalam kebijakan akses tidak akan berpengaruh. Perlu diketahui bahwa contoh kebijakan membatasi izin terhadap tindakan yang inklusif terjadi antara 1 Juli 2017 dan 31 Desember 2017 (UTC). Jika Anda masuk sebelum atau sesudah tanggal-tanggal tersebut, maka kebijakan tidak cocok, dan Anda tidak dapat mengasumsikan peran tersebut.

```
"Effect": "Allow",
"Principal": { "AWS": "arn:aws:iam::123456789012:root" },
"Action": "sts:AssumeRole",
"Condition": {
  "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
  "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
}
```

- Identitas Sumber — Administrator dapat mengonfigurasi peran untuk mewajibkan identitas untuk meneruskan string kustom yang mengidentifikasi orang atau aplikasi yang melakukan tindakan di AWS, yang disebut identitas sumber. Verifikasi apakah peran yang diambil mengharuskan identitas sumber ditetapkan. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Peran baru muncul di AWS akun

Beberapa AWS layanan mengharuskan Anda menggunakan jenis peran layanan unik yang ditautkan langsung ke layanan. Peran yang berkaitan dengan layanan ini telah ditetapkan sebelumnya oleh layanan dan mencakup semua izin yang diperlukan layanan. Hal ini memudahkan pengaturan layanan karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Untuk informasi umum tentang peran terkait layanan, lihat Buat peran tertaut layanan.

Anda mungkin sudah menggunakan layanan ketika layanan mulai mendukung peran yang terkait dengan layanan. Jika demikian, Anda mungkin menerima surel tentang peran baru di akun Anda. Peran ini mencakup semua izin yang diperlukan layanan untuk melakukan tindakan atas nama Anda. Anda tidak perlu mengambil tindakan apa pun untuk mendukung peran ini. Namun, Anda tidak boleh menghapus peran tersebut dari akun Anda. Melakukan hal tersebut dapat menghapus izin yang diperlukan layanan untuk mengakses sumber daya AWS . Anda dapat melihat peran terkait layanan di akun Anda dengan membuka halaman Peran IAM di konsol IAM. Service-linked peran muncul dengan (Service-linked peran) di kolom En titas tepercaya pada tabel.

Untuk informasi tentang layanan mana yang mendukung peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an. Untuk informasi tentang menggunakan peran yang terkait dengan layanan, pilih tautan Ya.

Saya tidak dapat mengedit atau menghapus peran di Akun AWS

Anda tidak dapat menghapus atau mengedit izin untuk peran yang terkait dengan layanan di IAM. Peran ini mencakup kepercayaan dan izin yang ditetapkan sebelumnya yang diperlukan oleh layanan untuk melakukan tindakan atas nama Anda. Anda dapat menggunakan konsol IAM, AWS CLI, atau API untuk mengedit hanya deskripsi peran terkait layanan. Anda dapat melihat peran terkait layanan di akun Anda dengan membuka halaman Peran IAM di konsol. Service-linked peran muncul dengan (Service-linked peran) di kolom En titas tepercaya pada tabel. Banner pada halaman Ringkasan peran juga menunjukkan bahwa peran tersebut adalah peran yang terkait dengan layanan. Anda dapat mengelola dan menghapus peran ini hanya melalui layanan yang terhubung, jika layanan tersebut mendukung tindakan. Berhati-hatilah saat memodifikasi atau menghapus peran terkait layanan karena melakukan hal ini dapat menghapus izin yang diperlukan layanan untuk mengakses sumber daya AWS .

Untuk informasi tentang layanan mana yang mendukung peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an.

Saya tidak berwenang untuk melakukan: iam: PassRole

Saat Anda membuat peran yang terkait dengan layanan, Anda harus memiliki izin untuk meneruskan peran tersebut ke layanan. Beberapa layanan secara otomatis membuat peran yang terkait dengan layanan di akun Anda ketika Anda melakukan tindakan di layanan tersebut. Misalnya, Amazon EC2 Auto Scaling membuat `AWSServiceRoleForAutoScaling` peran yang terkait dengan layanan untuk Anda pertama kali saat Anda membuat kelompok Auto Scaling. Jika Anda mencoba membuat grup Auto Scaling tanpa izin `PassRole`, Anda menerima kesalahan berikut ini:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Untuk memperbaiki kesalahan ini, minta administrator Anda untuk menambahkan izin `iam:PassRole` untuk Anda.

Untuk mempelajari layanan mana yang mendukung peran yang terkait dengan layanan, lihat AWS layanan yang bekerja dengan IAM. Untuk mempelajari apakah layanan secara otomatis membuat peran yang terkait dengan layanan untuk Anda, pilih tautan Ya untuk melihat dokumentasi peran layanan yang terkait dengan layanan.

Mengapa saya tidak bisa mengambil peran dengan sesi 12 jam? (AWS CLI, AWS API)

Saat Anda menggunakan operasi AWS STS `AssumeRole*` API atau `assume-role*` CLI untuk mengambil peran, Anda dapat menentukan nilai untuk `DurationSeconds` parameter. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan Durasi sesi maksimum untuk peran tersebut. Jika Anda menentukan nilai yang lebih tinggi dari pengaturan ini, operasi akan gagal. Pengaturan ini dapat memiliki nilai maksimum 12 jam. Misalnya, jika Anda menentukan durasi sesi 12 jam, tetapi administrator Anda mengatur durasi sesi maksimum menjadi 6 jam, operasi Anda gagal. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Jika Anda menggunakan rantai peran (menggunakan peran untuk mengsumsikan peran kedua), sesi Anda dibatasi hingga maksimum satu jam. Jika Anda kemudian menggunakan parameter `DurationSeconds` untuk memberikan nilai lebih dari satu jam, operasi akan gagal.

Saya menerima kesalahan saat mencoba beralih peran di konsol IAM

Informasi yang Anda masukkan di laman Beralih Peran harus sesuai dengan informasi untuk peran. Jika tidak, operasi akan gagal dan Anda menerima kesalahan berikut:

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Jika Anda menerima pesan kesalahan ini, pastikan bahwa informasi berikut benar:

- ID akun atau alias — Akun AWS ID adalah nomor 12 digit. Akun Anda mungkin memiliki alias, yang merupakan pengenal ramah seperti nama perusahaan Anda yang dapat digunakan sebagai pengganti Akun AWS ID Anda. Anda dapat menggunakan ID akun atau alias di bidang ini.
- Nama peran – Nama peran bersifat peka terhadap kapitalisasi huruf. ID akun dan nama peran harus sesuai dengan apa yang dikonfigurasi untuk peran tersebut.

Jika Anda terus menerima pesan kesalahan, hubungi administrator Anda untuk memverifikasi informasi sebelumnya. Kebijakan kepercayaan peran atau kebijakan pengguna IAM mungkin membatasi akses Anda. Administrator Anda dapat memverifikasi izin untuk kebijakan ini.

Peran saya memiliki kebijakan yang memungkinkan saya melakukan tindakan, tetapi saya mendapatkan “akses ditolak”

Sesi peran Anda mungkin dibatasi oleh kebijakan sesi. Bila Anda meminta kredentif keamanan sementara secara terprogram menggunakan AWS STS, Anda dapat meneruskan kebijakan sesi sebaris atau terkelola secara opsional. Kebijakan sesi adalah kebijakan lanjutan yang Anda sampaikan sebagai parameter saat Anda secara terprogram membuat sesi kredensial sementara untuk peran. Anda dapat menyampaikan satu dokumen kebijakan sesi inline JSON menggunakan parameter `Policy`. Anda dapat menggunakan parameter `PolicyArns` untuk menentukan hingga 10 kebijakan sesi terkelola. Izin sesi yang dihasilkan adalah persimpangan kebijakan berbasis identitas dan kebijakan sesi peran. Selain itu, jika administrator Anda atau program khusus memberi Anda kredensial sementara, mereka mungkin telah menyertakan kebijakan sesi untuk membatasi akses Anda.

Layanan tidak membuat versi kebijakan default peran tersebut

Peran layanan adalah peran yang diasumsikan oleh layanan untuk melakukan tindakan di akun Anda atas nama Anda. Ketika Anda mengatur beberapa lingkungan AWS layanan, Anda harus menentukan peran untuk diasumsikan oleh layanan. Dalam beberapa kasus, layanan tersebut menciptakan peran layanan dan kebijakannya di IAM untuk Anda. Meskipun Anda dapat mengubah atau menghapus peran layanan dan kebijakannya dari dalam IAM, ini AWS tidak disarankan. Peran dan kebijakan ditujukan hanya untuk digunakan oleh layanan tersebut. Jika Anda mengedit kebijakan dan menyiapkan lingkungan lain, ketika layanan mencoba menggunakan peran dan kebijakan yang sama, operasi dapat gagal.

Misalnya, saat Anda menggunakan AWS CodeBuild untuk pertama kalinya, layanan membuat peran bernama`codebuild-RWBCore-service-role`. Peran layanan tersebut menggunakan kebijakan bernama `codebuild-RWBCore-managed-policy`. Jika Anda mengedit kebijakan, ini akan membuat versi baru dan menyimpan versi tersebut sebagai versi default. Jika Anda melakukan operasi berikutnya di AWS CodeBuild, layanan mungkin mencoba memperbarui kebijakan. Jika iya, Anda akan menerima kesalahan berikut:

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
```

```
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

Jika Anda menerima kesalahan ini, Anda harus membuat perubahan di IAM sebelum Anda dapat melanjutkan dengan operasi layanan Anda. Pertama, atur versi kebijakan default ke V1 dan coba kembali operasi tersebut. Jika V1 sebelumnya dihapus, atau jika memilih V1 tidak berfungsi, maka bersihkan dan hapus kebijakan dan peran yang ada.

Untuk informasi selengkapnya tentang mengedit kebijakan terkelola, lihat Menyunting kebijakan terkelola pelanggan (konsol). Untuk informasi selengkapnya tentang versi kebijakan, lihat Peningkatan versi IAM.

Untuk menghapus peran layanan dan kebijakannya

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih nama kebijakan yang ingin Anda hapus.
4. Pilih tab En titas yang dilampirkan untuk melihat pengguna, grup, atau peran IAM mana yang menggunakan kebijakan ini. Jika salah satu dari identitas ini menggunakan kebijakan, selesaikan tugas berikut:
    a. Buat kebijakan terkelola baru dengan izin yang diperlukan. Untuk memastikan bahwa identitas memiliki izin yang sama sebelum dan setelah tindakan Anda, salin dokumen kebijakan JSON dari kebijakan yang ada. Kemudian buat kebijakan terkelola baru dan tempel dokumen JSON seperti yang dijelaskan dalam Membuat Kebijakan menggunakan editor JSON.
    b. Untuk setiap identitas yang terpengaruh, lampirkan kebijakan baru dan lepaskan kebijakan lama. Untuk informasi selengkapnya, lihat Menambahkan dan menghapus izin identitas IAM.
5. Di panel navigasi, pilih Peran.
6. Dalam daftar peran, pilih nama peran yang ingin Anda hapus.
7. Pilih tab Hubungan kepercayaan untuk melihat entitas mana yang dapat mengasumsikan peran tersebut. Jika entitas selain layanan dicantumkan, selesaikan tugas berikut:
    a. Buat peran baru yang memercayai entitas tersebut.
    b. Kebijakan yang Anda buat pada langkah sebelumnya. Jika Anda melewatkan langkah tersebut, buat kebijakan terkelola baru sekarang.

c. Beri tahu siapa pun yang mengasumsikan peran tersebut bahwa mereka tidak dapat lagi melakukannya. Berikan informasi kepada mereka tentang cara mengasumsikan peran baru dan memiliki izin yang sama.

8. Hapus kebijakan.
9. Hapus peran.

Tidak ada kasus penggunaan untuk peran layanan di konsol

Beberapa layanan mengharuskan Anda membuat peran layanan secara manual untuk memberi layanan izin untuk melakukan tindakan atas nama Anda. Jika layanan tidak terdaftar di konsol IAM, Anda harus mencantumkan layanan secara manual sebagai prinsipal tepercaya. Jika dokumentasi untuk layanan atau fitur yang Anda gunakan tidak menyertakan petunjuk untuk mencantumkan layanan sebagai prinsipal tepercaya, berikan umpan balik untuk halaman.

Untuk membuat peran layanan secara manual, Anda harus mengetahui prinsipal layanan untuk layanan yang akan mengambil peran. Sebuah prinsipal layanan adalah pengidentifikasi yang digunakan untuk memberikan izin layanan. Prinsipal layanan ditentukan oleh layanan tersebut.

Anda dapat menemukan prinsipal layanan untuk beberapa layanan dengan memeriksa hal berikut:

1. Buka AWS layanan yang bekerja dengan IAM.
2. Periksa apakah layanan memiliki Ya di kolom Service-linked peran.
3. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.
4. Temukan bagian izin Service-linked peran untuk layanan tersebut untuk melihat prinsi pal layanan.

Anda dapat secara manual membuat peran layanan menggunakan perintah AWS CLI atau operasi API AWS. Untuk membuat peran layanan secara manual menggunakan konsol IAM, selesaikan tugas berikut:

1. Buat peran IAM menggunakan ID akun Anda. Jangan lampirkan kebijakan atau beri izin apa pun. Lihat perinciannya di Buat peran untuk memberikan izin kepada pengguna IAM.
2. Buka peran dan edit hubungan kepercayaan. Alih-alih memercayai akun, peran harus memercayai layanan. Misalnya, perbarui yang elemen `Principal` berikut:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Ubah prinsip ke nilai untuk layanan Anda, seperti IAM.

```
"Principal": { "Service": "iam.amazonaws.com" }
```

3. Tambahkan izin yang diperlukan layanan dengan melampirkan kebijakan izin untuk peran.
4. Kembali ke layanan yang memerlukan izin dan gunakan metode yang didokumentasikan untuk memberi tahu layanan tentang peran layanan baru.

Memecahkan masalah IAM dan Amazon EC2

Informasi berikut dapat membantu Anda memecahkan masalah IAM dengan Amazon EC2.

Topik

- Ketika saya mencoba meluncurkan instance, saya tidak melihat peran dalam daftar Peran IAM konsol Amazon EC2
- Kredensial pada instans saya untuk peran yang salah
- Saat saya mencoba memanggil AddRoleToInstanceProfile, saya mendapatkan kesalahan AccessDenied
- Amazon EC2: Ketika saya mencoba meluncurkan instance dengan peran, saya mendapatkan kesalahan AccessDenied
- Saya tidak dapat mengakses kredensial keamanan sementara di instans EC2 saya
- Apa arti kesalahan dari dokumen info di subtree IAM?

Ketika saya mencoba meluncurkan instance, saya tidak melihat peran dalam daftar Peran IAM konsol Amazon EC2

Periksa hal-hal berikut:

- Jika Anda masuk sebagai pengguna IAM, pastikan Anda memiliki izin untuk memanggil`ListInstanceProfiles`. Untuk informasi tentang izin yang diperlukan untuk bekerja dengan peran, lihatIzin yang diperlukan untuk menggunakan peran dengan Amazon EC2. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola kebijakan IAM.

  Jika Anda tidak dapat mengubah izin Anda sendiri, Anda harus menghubungi administrator yang dapat bekerja dengan IAM untuk memperbarui izin Anda.

- Jika Anda membuat peran menggunakan IAM CLI atau API, verifikasi hal berikut:
  - Anda membuat profil instance dan menambahkan peran ke profil instance tersebut.
  - Anda menggunakan nama yang sama untuk peran dan profil instance. Jika Anda memberi nama profil peran dan instans secara berbeda, Anda tidak akan melihat nama peran yang benar di konsol Amazon EC2.

  Daftar Peran IAM di konsol Amazon EC2 mencantumkan nama profil instans, bukan nama peran. Anda harus memilih nama profil instans yang berisi peran yang Anda inginkan. Untuk detail tentang profil instans, lihat Gunakan profil instance.

  > **Note**
  >
  > Jika Anda menggunakan konsol IAM untuk membuat peran, Anda tidak perlu bekerja dengan profil instans. Untuk setiap peran yang Anda buat di konsol IAM, profil instans dibuat dengan nama yang sama dengan peran tersebut, dan peran tersebut secara otomatis ditambahkan ke profil instans tersebut. Sebuah profil instans hanya dapat berisi satu peran IAM, dan batas itu tidak dapat ditingkatkan.

Kredensial pada instans saya untuk peran yang salah

Peran dalam profil instans mungkin telah diganti baru-baru ini. Jika demikian, aplikasi Anda perlu menunggu rotasi kredensial terjadwal otomatis berikutnya sebelum kredensial untuk peran Anda tersedia.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Saat saya mencoba memanggil **`AddRoleToInstanceProfile`**, saya mendapatkan kesalahan **`AccessDenied`**

Jika Anda membuat permintaan sebagai pengguna IAM, verifikasi bahwa Anda memiliki izin berikut:

- `iam:AddRoleToInstanceProfile` dengan sumber daya yang cocok dengan ARN profil instans (misalnya, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Untuk informasi selengkapnya tentang izin yang diperlukan untuk bekerja dengan peran, lihatBagaimana saya memulainya?. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola kebijakan IAM.

Amazon EC2: Ketika saya mencoba meluncurkan instance dengan peran, saya mendapatkan kesalahan **`AccessDenied`**

Periksa hal-hal berikut:

- Luncurkan instans tanpa profil instans. Hal ini akan membantu memastikan bahwa masalah terbatas pada peran IAM untuk instans Amazon EC2.
- Jika Anda membuat permintaan sebagai pengguna IAM, verifikasi bahwa Anda memiliki izin berikut:
  - `ec2:RunInstances` dengan sumber daya wildcard ("*")
  - `iam:PassRole` dengan sumber daya yang cocok dengan ARN peran (misalnya, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Panggil tindakan `GetInstanceProfile` IAM untuk memastikan bahwa Anda menggunakan nama profil instans valid atau ARN profil instans valid. Untuk informasi lebih lanjut, lihat Menggunakan peran IAM dengan instans Amazon EC2 .
- Panggil tindakan `GetInstanceProfile` IAM untuk memastikan profil instans memiliki peran. Profil instans kosong akan gagal dengan kesalahan `AccessDenied`. Untuk informasi selengkapnya tentang membuat peran, lihat Pembuatan peran IAM.

Untuk informasi selengkapnya tentang izin yang diperlukan untuk bekerja dengan peran, lihatBagaimana saya memulainya?. Untuk informasi tentang menambahkan izin ke pengguna, lihat Kelola kebijakan IAM.

Saya tidak dapat mengakses kredensial keamanan sementara di instans EC2 saya

Untuk mengakses kredensial keamanan sementara pada instans EC2 Anda, Anda harus menggunakan konsol IAM terlebih dahulu untuk membuat peran. Kemudian Anda meluncurkan instans EC2 yang menggunakan peran tersebut dan memeriksa instans yang berjalan. Untuk informasi selengkapnya, lihat Bagaimana Cara Memulai? di Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2.

Jika Anda masih tidak dapat mengakses kredensial keamanan sementara Anda di instans EC2, periksa hal berikut:

- Bisakah Anda mengakses bagian lain dari Layanan Metadata Instans (IMDS)? Jika tidak, periksa bahwa Anda tidak memiliki aturan firewall yang memblokir akses permintaan ke IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- Apakah ada subtree `iam` IMDS? Jika tidak, verifikasi bahwa instans Anda memiliki profil instans IAM yang terkait dengannya dengan memanggil operasi EC2 `DescribeInstances` API atau menggunakan perintah CLI`aws ec2 describe-instances`.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Periksa kesalahan pada dokumen `info` di subtree IAM. Jika Anda memiliki kesalahan, lihat Apa arti kesalahan dari dokumen `info` di subtree IAM? untuk informasi selengkapnya.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

Apa arti kesalahan dari dokumen **`info`** di subtree IAM?

Dokumen **`iam/info`** berarti **`"Code":"InstanceProfileNotFound"`**

Profil instans IAM Anda telah dihapus dan Amazon EC2 tidak dapat lagi memberikan kredensial ke instans Anda. Anda harus melampirkan profil instans yang valid ke instans Amazon EC2 Anda.

Jika ada profil instans dengan nama tersebut, periksa apakah profil instans tidak dihapus dan penggantinya dibuat dengan nama yang sama:

1. Panggil operasi `GetInstanceProfile` IAM untuk mendapatkan `InstanceProfileId`.
2. Panggil operasi `DescribeInstances` Amazon EC2 untuk mendapatkan `IamInstanceProfileId` untuk instans.
3. Verifikasi bahwa `InstanceProfileId` dari operasi IAM sesuai dengan `IamInstanceProfileId` dari operasi Amazon EC2.

Jika IDs berbeda, maka profil instance yang dilampirkan ke instance Anda tidak lagi valid. Anda harus melampirkan profil instans yang valid ke instans.

Dokumen **`iam/info`** menunjukkan keberhasilan tetapi juga menunjukkan **`"Message":"Instance Profile does not contain a role..."`**

Peran tersebut telah dihapus dari profil instans oleh tindakan `RemoveRoleFromInstanceProfile` IAM. Anda dapat menggunakan tindakan `AddRoleToInstanceProfile` IAM untuk melampirkan peran ke profil instans. Aplikasi Anda perlu menunggu hingga penyegaran yang terjadwal berikutnya untuk mengakses kredensial untuk peran tersebut.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Dokumen **`iam/security-credentials/[role-name]`** menunjukkan **`"Code":"AssumeRoleUnauthorizedAccess"`**

Amazon EC2 tidak memiliki izin untuk mengambil peran tersebut. Izin untuk mengambil peran tersebut dikendalikan oleh kebijakan kepercayaan yang terlampir pada peran tersebut, seperti contoh berikut. Gunakan API `UpdateAssumeRolePolicy` IAM untuk memperbarui kebijakan kepercayaan.

JSON

```
{"Version":"2012-10-17",      "Statement": [{"Effect": "Allow","Principal":
 {"Service": ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

Aplikasi Anda perlu menunggu hingga penyegaran terjadwal otomatis berikutnya untuk mengakses kredensial untuk peran tersebut.

Untuk memaksa perubahan, Anda harus memisahkan profil instans dan kemudian mengaitkan profil instans, atau Anda dapat menghentikan instance Anda lalu memulainya ulang.

Memecahkan masalah IAM dan Amazon S3

Gunakan informasi di sini untuk membantu Anda memecahkan dan memperbaiki masalah yang mungkin Anda hadapi saat bekerja dengan Amazon S3 dan IAM.

Bagaimana saya memberikan akses anonim ke bucket Amazon S3?

Anda menggunakan kebijakan bucket Amazon S3 yang menentukan wildcard (*) di elemen `principal`, yang berarti siapa pun dapat mengakses bucket. Dengan akses anonim, siapa pun (termasuk pengguna tanpa Akun AWS) akan dapat mengakses bucket. Untuk kebijakan sampel, lihat Contoh Kasus untuk Kebijakan Bucket Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Saya masuk sebagai Akun AWS pengguna root. Mengapa saya tidak dapat mengakses bucket Amazon S3 di bawah akun saya?

Dalam beberapa kasus, Anda mungkin memiliki pengguna IAM dengan akses penuh ke IAM dan Amazon S3. Jika pengguna IAM menetapkan kebijakan bucket ke bucket Amazon S3 dan tidak menentukan pengguna root sebagai prinsipal, pengguna root ditolak akses ke bucket tersebut. Namun, sebagai pengguna utama, Anda masih dapat mengakses bucket. Untuk melakukannya, ubah kebijakan bucket untuk mengizinkan akses pengguna root dari konsol Amazon S3 atau AWS CLI. Gunakan prinsip berikut, ganti *123456789012* dengan ID Akun AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Memecahkan masalah federasi SAML dengan IAM

Gunakan informasi di sini untuk membantu Anda mendiagnosis dan memperbaiki masalah yang mungkin Anda temukan saat bekerja dengan SAML 2.0 dan federasi dengan AWS Identity and Access Management.

Topik

- Kesalahan: Permintaan Anda menyertakan respons SAML yang tidak valid. Untuk keluar, klik di sini.
- Kesalahan: RoleSessionName diperlukan dalam AuthnResponse (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)
- Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWithSAML (layanan: AWSSecurityTokenService; kode status: 403; kode kesalahan: AccessDenied)
- Kesalahan: RoleSessionName dalam AuthnResponse harus cocok [a-z A-Z _0-9+=, .@-] {2,64} (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan:) InvalidIdentityToken

- Kesalahan: Identitas Sumber harus cocok dengan [a-z A-Z _0-9+=, .@-] {2,256} dan tidak dimulai dengan “aws:” (layanan:; kode status: 400 AWSSecurityTokenService; kode kesalahan:) InvalidIdentityToken
- Kesalahan: Tanda tangan respons tidak valid (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)
- Kesalahan: Kunci pribadi tidak valid.
- Kesalahan: Gagal menghapus kunci pribadi.
- Kesalahan: Gagal menghapus kunci pribadi karena ID Kunci tidak cocok dengan kunci pribadi.
- Kesalahan: Gagal mengambil peran: Penerbit tidak ada di penyedia tertentu (layanan: AWSOpenIdDiscoveryService; kode status: 400; kode kesalahan: AuthSamlInvalidSamlResponseException)
- Kesalahan: Tidak dapat mengurai metadata.
- Kesalahan: Tidak dapat memperbarui penyedia identitas. Tidak ada pembaruan yang ditentukan untuk metadata atau pernyataan enkripsi.
- Kesalahan: Tidak dapat mengatur mode enkripsi pernyataan ke Diperlukan karena tidak ada kunci pribadi yang disediakan.
- Kesalahan: Tidak dapat menambah dan menghapus kunci pribadi dalam permintaan yang sama. Tetapkan nilai untuk hanya satu dari dua parameter.
- Kesalahan: Penyedia yang ditentukan tidak ada.
- Kesalahan: D DurationSeconds iminta melebihi MaxSessionDuration set untuk peran ini.
- Kesalahan: Batas kunci pribadi 2 tercapai.
- Kesalahan: Respons tidak berisi audiens yang diperlukan.

Kesalahan: Permintaan Anda menyertakan respons SAML yang tidak valid. Untuk keluar, klik di sini.

Kesalahan ini dapat terjadi saat respons SAML dari penyedia identitas tidak menyertakan atribut dengan `Name` yang ditetapkan ke `https://aws.amazon.com/SAML/Attributes/Role`. Atribut harus berisi satu atau beberapa elemen `AttributeValue`, masing-masing berisi sepasang string yang dipisahkan koma:

- ARN peran yang tempat pengguna dapat dipetakan
- ARN penyedia SAML

Kesalahan juga dapat terjadi ketika nilai atribut SAML yang dikirim oleh penyedia identitas memiliki spasi awal atau akhir, atau karakter tidak valid lainnya dalam nilai atribut SAML. Untuk informasi selengkapnya tentang nilai yang diharapkan untuk atribut SAML, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi

Untuk informasi selengkapnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di Melihat respons SAML di browser Anda.

Kesalahan: RoleSessionName diperlukan dalam AuthnResponse (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)

Kesalahan ini dapat terjadi saat respons SAML dari penyedia identitas tidak menyertakan atribut dengan `Name` yang ditetapkan ke `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. Nilai atribut adalah pengidentifikasi untuk pengguna dan umumnya adalah ID pengguna atau alamat email.

Untuk informasi selengkapnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di Melihat respons SAML di browser Anda.

Kesalahan: Tidak berwenang untuk melakukan sts: AssumeRoleWithSAML (layanan: AWSSecurityTokenService; kode status: 403; kode kesalahan: AccessDenied)

Kesalahan ini dapat terjadi jika IAM role yang ditentukan dalam respons SAML salah eja atau tidak ada. Pastikan untuk menggunakan nama yang tepat dari peran Anda, karena nama peran bersifat peka kapitalisasi huruf. Perbaiki nama peran dalam konfigurasi penyedia layanan SAML.

Anda diperbolehkan mengakses hanya jika kebijakan kepercayaan peran Anda mencakup tindakan `sts:AssumeRoleWithSAML`. Jika pernyataan SAML Anda dikonfigurasikan untuk menggunakan atribut `PrincipalTag`, kebijakan kepercayaan Anda juga harus mencakup tindakan `sts:TagSession`. Untuk informasi selengkapnya tentang tanda sesi, lihat Lewati tag sesi di AWS STS.

Kesalahan ini dapat terjadi jika Anda tidak memiliki izin `sts:SetSourceIdentity` dalam kebijakan kepercayaan peran Anda. Jika pernyataan SAML Anda dikonfigurasikan untuk

menggunakan atribut `SourceIdentity`, kebijakan kepercayaan Anda juga harus mencakup tindakan `sts:SetSourceIdentity`. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Kesalahan ini juga dapat terjadi jika kepala sekolah federasi tidak memiliki izin untuk mengambil peran tersebut. Peran harus memiliki kebijakan kepercayaan yang menentukan ARN penyedia identitas IAM SAML sebagai. `Principal` Peran ini juga berisi kondisi yang mengontrol pengguna mana yang dapat mengasumsikan peran tersebut. Pastikan bahwa pengguna Anda memenuhi persyaratan kondisi.

Kesalahan ini juga dapat terjadi jika respons SAML tidak mencakup `Subject` yang mengandung `NameID`.

Untuk informasi selengkapnya, lihat Mengaktifkan prinsipal federasi SAMP 2.0 untuk mengakses Konsol Manajemen AWS dan Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di Melihat respons SAML di browser Anda.

Kesalahan: RoleSessionName dalam AuthnResponse harus cocok [a-z A-Z _0-9+=, .@-] {2,64} (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan:) InvalidIdentityToken

Kesalahan ini dapat terjadi jika nilai atribut `RoleSessionName` terlalu panjang atau mengandung karakter yang tidak valid. Panjang valid maksimum adalah 64 karakter.

Untuk informasi selengkapnya, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di Melihat respons SAML di browser Anda.

Kesalahan: Identitas Sumber harus cocok dengan [a-z A-Z _0-9+=, .@-] {2,256} dan tidak dimulai dengan **“aws:”** (layanan:; kode status: 400 AWSSecurityTokenService; kode kesalahan:) InvalidIdentityToken

Kesalahan ini dapat terjadi jika nilai atribut `sourceIdentity` terlalu panjang atau mengandung karakter yang tidak valid. Panjang maksimum yang valid adalah 256 karakter. Untuk informasi selengkapnya tentang identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

Untuk informasi selengkapnya tentang cara membuat pernyataan SAML, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi. Untuk melihat respons SAML di peramban Anda, ikuti langkah-langkah yang tercantum di Melihat respons SAML di browser Anda.

Kesalahan: Tanda tangan respons tidak valid (layanan: AWSSecurityTokenService; kode status: 400; kode kesalahan: InvalidIdentityToken)

Kesalahan ini dapat terjadi saat metadata federasi penyedia identitas tidak cocok dengan metadata penyedia identitas IAM. Misalnya, file metadata untuk penyedia layanan identitas mungkin telah berubah untuk memperbarui sertifikat yang kedaluwarsa. Unduh file metadata SAML yang diperbarui dari penyedia layanan identitas Anda. Kemudian perbarui di entitas penyedia AWS identitas yang Anda tentukan di IAM dengan perintah CLI `aws iam update-saml-provider` lintas platform atau cmdlet. `Update-IAMSAMLProvider` PowerShell

Kesalahan: Kunci pribadi tidak valid.

Kesalahan ini dapat terjadi jika Anda tidak memformat file kunci pribadi Anda dengan benar. Kesalahan ini dapat memberikan detail tambahan tentang mengapa kunci pribadi tidak valid:

- Kunci dienkripsi.
- Format kunci tidak dikenali. File kunci pribadi harus berkas.pem.

Saat Anda Buat penyedia identitas SAML di IAM masuk Konsol Manajemen AWS, Anda harus mengunduh kunci pribadi dari penyedia identitas Anda untuk diberikan kepada IAM untuk mengaktifkan enkripsi. Kunci pribadi harus berupa file.pem yang menggunakan AES-GCM atau algoritma AES-CBC enkripsi untuk mendekripsi pernyataan SAML.

Kesalahan: Gagal menghapus kunci pribadi.

Kesalahan ini dapat terjadi ketika enkripsi SAML diatur ke Diperlukan, dan permintaan Anda akan menghapus satu-satunya kunci dekripsi pribadi untuk penyedia IAM SAML. Untuk informasi selengkapnya tentang memutar kunci pribadi, lihatKelola kunci enkripsi SAML.

Kesalahan: Gagal menghapus kunci pribadi karena ID Kunci tidak cocok dengan kunci pribadi.

Kesalahan ini dapat terjadi jika `keyId` nilai kunci pribadi tidak cocok dengan salah satu ID Kunci untuk file kunci pribadi penyedia identitas.

Saat Anda menggunakan operasi update-saml-provider atau UpdateSamlProvider API untuk menghapus kunci pribadi enkripsi SAML, nilai di dalam `RemovePrivateKey` harus berupa ID Kunci yang valid untuk kunci pribadi yang dilampirkan ke penyedia identitas Anda.

Kesalahan: Gagal mengambil peran: Penerbit tidak ada di penyedia tertentu (layanan: AWSOpenIdDiscoveryService; kode status: 400; kode kesalahan: AuthSamlInvalidSamlResponseException)

Kesalahan ini dapat terjadi jika penerbit dalam respons SAML tidak cocok dengan penerbit yang dinyatakan dalam file metadata federasi. File metadata diunggah ke AWS saat Anda membuat penyedia identitas di IAM.

Kesalahan: Tidak dapat mengurai metadata.

Kesalahan ini dapat terjadi jika Anda tidak memformat file metadata Anda dengan benar.

Saat Anda membuat atau mengelola penyedia identitas SAML di Konsol Manajemen AWS, Anda harus mengambil dokumen metadata SAML dari penyedia identitas Anda.

File metadata ini mencakup nama penerbit, informasi kedaluwarsa, dan kunci yang dapat digunakan untuk memvalidasi respons otentikasi SAML (pernyataan) yang diterima dari IdP. File metadata harus dikodekan dalam UTF-8 format tanpa tanda urutan byte (BOM). Untuk menghapus BOM, Anda dapat menyandikan file seperti UTF-8 menggunakan alat pengeditan teks, seperti Notepad ++.

X.509 Sertifikat yang disertakan sebagai bagian dari dokumen metadata SAML harus menggunakan ukuran kunci minimal 1024 bit. Selain itu, X.509 sertifikat juga harus bebas dari ekstensi berulang. Anda dapat menggunakan ekstensi, tetapi ekstensi hanya dapat muncul sekali di sertifikat. Jika X.509 sertifikat tidak memenuhi salah satu kondisi, pembuatan IdP gagal dan mengembalikan kesalahan “Tidak dapat mengurai metadata”.

Sebagaimana didefinisikan oleh Profil Interoper V2.0 abilitas Metadata SAML Versi 1.0, IAM tidak mengevaluasi atau mengambil tindakan pada kedaluwarsa X.509 sertifikat dalam dokumen metadata

SAML. Jika Anda khawatir tentang X.509 sertifikat yang kedaluwarsa, sebaiknya pantau tanggal kedaluwarsa sertifikat dan merotasi sertifikat sesuai dengan kebijakan tata kelola dan keamanan organisasi Anda.

Kesalahan: Tidak dapat memperbarui penyedia identitas. Tidak ada pembaruan yang ditentukan untuk metadata atau pernyataan enkripsi.

Kesalahan ini dapat terjadi jika Anda menggunakan operasi `update-saml-provider` CLI atau `UpdateSAMLProvider` API, tetapi tidak memberikan nilai pembaruan dalam parameter permintaan Anda. Untuk informasi selengkapnya tentang memperbarui penyedia IAM SAML Anda, lihat. Buat penyedia identitas SAML di IAM

Kesalahan: Tidak dapat mengatur mode enkripsi pernyataan ke Diperlukan karena tidak ada kunci pribadi yang disediakan.

Kesalahan ini dapat terjadi ketika Anda belum mengunggah kunci dekripsi pribadi sebelumnya dan Anda mencoba mengatur enkripsi SAML ke Diperlukan tanpa menyertakan kunci pribadi dalam permintaan Anda.

Pastikan kunci pribadi ditentukan untuk penyedia IAM SAML Anda saat menggunakan operasi CLI, API, `create-saml-provider` `update-saml-provider` CLI, atau `CreateSAMLProvider` `UpdateSAMLProvider` API untuk memerlukan pernyataan SAML terenkripsi.

Kesalahan: Tidak dapat menambah dan menghapus kunci pribadi dalam permintaan yang sama. Tetapkan nilai untuk hanya satu dari dua parameter.

Kesalahan ini dapat terjadi jika nilai tambah dan hapus kunci pribadi disertakan dalam permintaan yang sama.

Saat menggunakan operasi update-saml-provider atau UpdateSamlProvider API untuk memutar file kunci pribadi enkripsi SAML, Anda hanya dapat menambahkan atau menghapus kunci pribadi dalam permintaan Anda. Jika Anda menambahkan kunci pribadi saat menghapus kunci pribadi, operasi gagal. Untuk informasi selengkapnya tentang memutar kunci pribadi, lihatKelola kunci enkripsi SAML.

Kesalahan: Penyedia yang ditentukan tidak ada.

Kesalahan ini dapat terjadi jika nama penyedia dalam pernyataan SAML tidak cocok dengan nama penyedia di IAM. Untuk informasi selengkapnya tentang melihat nama penyedia, lihat Buat penyedia identitas SAML di IAM.

Kesalahan: D DurationSeconds iminta melebihi MaxSessionDuration set untuk peran ini.

Kesalahan ini dapat terjadi jika Anda mengambil peran dari API AWS CLI or.

Saat Anda menggunakan CLI assume-role-with-saml atau operasi API AssumeRoleWithSAML untuk mengambil peran, Anda dapat menentukan nilai untuk parameter `DurationSeconds`. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut. Jika Anda menentukan nilai yang lebih tinggi dari pengaturan ini, operasi gagal. Misalnya, jika Anda menentukan durasi sesi 12 jam, tetapi administrator Anda mengatur durasi sesi maksimum menjadi 6 jam, operasi Anda gagal. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Kesalahan: Batas kunci pribadi 2 tercapai.

Kesalahan ini dapat terjadi jika Anda mencoba menambahkan kunci pribadi ke penyedia identitas Anda.

Anda dapat menyimpan hingga dua kunci pribadi untuk setiap penyedia identitas. Bila Anda menggunakan operasi update-saml-provider atau up dateSamlProvider API untuk menambahkan kunci pribadi ketiga, operasi gagal.

Hapus kunci pribadi yang telah kedaluwarsa sebelum menambahkan kunci pribadi baru. Untuk informasi selengkapnya tentang memutar kunci pribadi, lihatKelola kunci enkripsi SAML.

Kesalahan: Respons tidak berisi audiens yang diperlukan.

Kesalahan ini dapat terjadi jika ada ketidakcocokan antara URL audiens dan penyedia identitas dalam konfigurasi SAML. Pastikan pengidentifikasi pihak yang mengandalkan penyedia identitas (IdP) Anda sama persis dengan URL audiens (ID entitas) yang disediakan dalam konfigurasi SAML.

Bagaimana IAM bekerja dengan layanan lain AWS

Sementara IAM adalah AWS layanan utama yang akan Anda gunakan untuk mengelola sumber daya IAM, semua AWS layanan lain bekerja dengan IAM untuk mengontrol akses ke sumber daya di akun Anda.

- CloudFormation

  CloudFormation terintegrasi dengan IAM dengan memungkinkan Anda untuk mendefinisikan dan mengelola sumber daya IAM sebagai bagian dari template Anda. CloudFormation Anda dapat menggunakan CloudFormation untuk menentukan izin IAM yang diperlukan untuk AWS sumber daya lain yang Anda berikan. CloudFormation juga mendukung penggunaan peran IAM untuk mengelola kredensional yang diperlukan untuk penyediaan dan pengelolaan AWS infrastruktur Anda, dan fitur deteksi driftnya membantu Anda menjaga integritas konfigurasi IAM Anda.
- AWS CloudShell

  Saat Anda mengakses AWS CloudShell, otentikasi dan otorisasi Anda ditangani melalui IAM. AWS CloudShell berjalan dalam konteks peran IAM yang ditetapkan ke pengguna atau akun Anda. Ketika Anda meluncurkan AWS CloudShell, secara otomatis menghasilkan kredenal keamanan sementara berdasarkan peran IAM yang ditetapkan untuk Anda.
- AWS SDKs

  AWS SDKs Bekerja dengan IAM dengan menangani proses otentikasi dan otorisasi, mengelola AWS kredensional, dan menghormati izin dan kebijakan yang ditentukan dalam IAM untuk memastikan aplikasi Anda hanya dapat mengakses sumber daya yang diizinkan untuk digunakan. SDKs Menyediakan mekanisme untuk memperoleh dan menggunakan kredenal keamanan sementara, serta memvalidasi izin yang diperlukan untuk operasi aplikasi Anda.

Untuk daftar AWS layanan yang bekerja dengan IAM dan IAM menampilkan dukungan layanan, lihat. AWS layanan yang bekerja dengan IAM

Buat sumber daya IAM dengan CloudFormation

AWS Identity and Access Management terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur AWS sumber daya Anda sehingga Anda dapat menghabiskan lebih sedikit waktu untuk membuat dan mengelola sumber daya dan infrastruktur

Anda. Anda membuat templat yang menjelaskan semua AWS sumber daya yang Anda inginkan (seperti kunci akses, grup, kebijakan grup, profil instans, kebijakan terkelola, penyedia OIDC, kebijakan sebaris, peran, kebijakan peran, penyedia SAMM, sertifikat server, peran terkait layanan, pengguna (dan menambahkan pengguna ke grup), kebijakan pengguna, dan perangkat MFA virtual), serta menyediakan serta mengonfigurasi sumber daya tersebut untuk Anda. CloudFormation

Bila Anda menggunakan CloudFormation, Anda dapat menggunakan kembali template Anda untuk mengatur sumber daya IAM Anda secara konsisten dan berulang kali. Jelaskan sumber daya Anda sekali, lalu sediakan sumber daya yang sama berulang-ulang di beberapa Akun AWS dan Wilayah.

IAM dan template CloudFormation

Untuk menyediakan dan mengonfigurasi sumber daya untuk IAM dan layanan terkait, Anda harus memahami CloudFormation templat. Templat adalah file teks dengan format JSON atau YAML. Template ini menjelaskan sumber daya yang ingin Anda sediakan di CloudFormation tumpukan Anda. Jika Anda tidak terbiasa dengan JSON atau YAMM, Anda dapat menggunakan CloudFormation Designer untuk membantu Anda memulai dengan template. CloudFormation Untuk informasi selengkapnya, lihat Apa itu CloudFormation Designer? di Panduan Pengguna AWS CloudFormation .

IAM mendukung pembuatan kunci akses, grup, kebijakan grup, profil instans, kebijakan terkelola, penyedia OIDC, kebijakan inline, peran, kebijakan peran, penyedia SAMM, sertifikat server, peran terkait layanan, pengguna (dan menambahkan pengguna ke grup), kebijakan pengguna, dan perangkat MFA virtual di. CloudFormationUntuk informasi selengkapnya, termasuk contoh template JSON dan YAMAL untuk sumber daya IAM, lihat referensi jenis AWS Identity and Access Management sumber daya di Panduan Pengguna.AWS CloudFormation

Anda juga dapat membuat templat yang membuat sumber daya terkait, seperti peran dan kebijakan terkelola.

Pelajari lebih lanjut tentang CloudFormation

Untuk mempelajari selengkapnya CloudFormation, lihat sumber daya berikut:

- AWS CloudFormation
- AWS CloudFormation Panduan Pengguna
- CloudFormation Referensi API
- AWS CloudFormation Panduan Pengguna Antarmuka Baris Perintah

Gunakan AWS CloudShell untuk bekerja dengan AWS Identity and Access Management

AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari. Konsol Manajemen AWS Anda dapat menjalankan AWS CLI perintah terhadap AWS layanan (termasuk AWS Identitas dan Manajemen Akses) menggunakan shell pilihan Anda (Bash, PowerShell atau Z shell). Anda juga dapat melakukan ini tanpa perlu mengunduh atau menginstal alat baris perintah.

Anda meluncurkan AWS CloudShell dari Konsol Manajemen AWS, dan AWS kredenSIAL yang Anda gunakan untuk masuk ke konsol secara otomatis tersedia di sesi shell baru. Pra-otentikasi AWS CloudShell pengguna ini memungkinkan Anda untuk melewati konfigurasi kredenSIAL saat berinteraksi dengan AWS layanan seperti IAM menggunakan AWS CLI versi 2 (pra-instal di lingkungan komputasi shell).

Dapatkan izin IAM untuk AWS CloudShell

Menggunakan sumber daya manajemen akses yang disediakan oleh AWS Identity and Access Management, administrator dapat memberikan izin kepada pengguna IAM sehingga mereka dapat mengakses AWS CloudShell dan menggunakan fitur lingkungan.

Cara tercepat bagi administrator untuk memberikan akses kepada pengguna adalah melalui kebijakan terkel AWS ola. Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. Kebijakan terkel AWS ola berikut untuk CloudShell dapat dilampirkan ke identitas IAM:

- `AWSCloudShellFullAccess`: Memberikan izin untuk digunakan AWS CloudShell dengan akses penuh ke semua fitur.

Jika Anda ingin membatasi ruang lingkup tindakan yang dapat dilakukan pengguna IAM AWS CloudShell, Anda dapat membuat kebijakan kustom yang menggunakan kebijakan terkel `AWSCloudShellFullAccess` ola sebagai templat. Untuk informasi selengkapnya tentang membatasi tindakan yang tersedia bagi pengguna CloudShell, lihat M engelola AWS CloudShell akses dan penggunaan dengan kebijakan IAM di Panduan AWS CloudShell Pengguna.

Berinteraksi dengan IAM

Setelah Anda meluncurkan AWS CloudShell dari Konsol Manajemen AWS, Anda dapat segera mulai berinteraksi dengan IAM menggunakan antarmuka baris perintah.

> **Note**
>
> Saat menggunakan AWS CLI in AWS CloudShell, Anda tidak perlu mengunduh atau menginstal sumber daya tambahan apa pun. Selain itu, karena Anda sudah diautentikasi di dalam shell, Anda tidak perlu mengonfigurasi kredenSIAL sebelum melakukan panggilan.

Buat grup IAM dan tambahkan pengguna IAM ke grup menggunakan AWS CloudShell

Contoh berikut digunakan CloudShell untuk membuat grup IAM, menambahkan pengguna IAM ke grup, dan kemudian memverifikasi bahwa perintah berhasil.

1. Dari Konsol Manajemen AWS, Anda dapat meluncurkan CloudShell dengan memilih opsi berikut yang tersedia di bilah navigasi:
   - Pilih CloudShell ikon.
   - Mulai ketik “cloudshell” di kotak Pencarian dan kemudian pilih CloudShell opsi.
2. Untuk membuat grup IAM, masukkan perintah berikut di baris CloudShell perintah. Dalam contoh ini kita menamai grup east_coast:

   ```
   aws iam create-group --group-name east_coast
   ```

   Jika panggilan berhasil, baris perintah menampilkan respons dari layanan yang mirip dengan output berikut:

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
          "CreateDate": "2023-09-11T21:02:21+00:00"
       }
   ```

```
        }
```

3. Untuk menambahkan pengguna ke grup yang Anda buat, gunakan perintah berikut, tentukan nama grup dan nama pengguna. Dalam contoh ini kita menamai grup east_coast dan user john:

```
aws iam add-user-to-group --group-name east_coast --user-name john
```

4. Untuk memverifikasi bahwa pengguna ada dalam grup, gunakan perintah berikut, tentukan nama grup. Dalam contoh ini kita terus menggunakan grup east_coast:

```
aws iam get-group --group-name east_coast
```

Jika panggilan berhasil, baris perintah menampilkan respons dari layanan yang mirip dengan output berikut:

```
      {
        "Users": [
          {
              "Path": "/",
              "UserName": "john",
              "UserId": "AIDAYBDBW4JBXGEXAMPLE",
              "Arn": "arn:aws:iam::552108220995:user/john",
              "CreateDate": "2023-09-11T20:43:14+00:00",
              "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
          }
        ],
        "Group": {
              "Path": "/",
              "GroupName": "east_coast",
              "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
              "Arn": "arn:aws:iam::111122223333:group/east_coast",
              "CreateDate": "2023-09-11T21:02:21+00:00"
           }
      }
```

Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
AWS SDK untuk C++	AWS SDK untuk C++ contoh kode
AWS CLI	AWS CLI contoh kode
AWS SDK untuk Go	AWS SDK untuk Go contoh kode
AWS SDK untuk Java	AWS SDK untuk Java contoh kode
AWS SDK untuk JavaScript	AWS SDK untuk JavaScript contoh kode
AWS SDK untuk Kotlin	AWS SDK untuk Kotlin contoh kode
AWS SDK untuk .NET	AWS SDK untuk .NET contoh kode
AWS SDK untuk PHP	AWS SDK untuk PHP contoh kode
Alat AWS untuk PowerShell	Alat AWS untuk PowerShell contoh kode
AWS SDK untuk Python v4	AWS Contoh kode SDK untuk Python v4
AWS SDK untuk Python (Boto3)	AWS SDK untuk Python (Boto3) contoh kode
AWS SDK untuk Ruby	AWS SDK untuk Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk layanan ini, lihat Contoh kode untuk IAM menggunakan AWS SDK.

Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Informasi referensi untuk AWS Identity and Access Management

Gunakan topik di bagian ini untuk menemukan bahan referensi terperinci untuk berbagai aspek IAM dan AWS STS.

Topik

- Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARN)
- Pengidentifikasi IAM
- IAM dan AWS STS kuota
- Dukungan titik akhir tumpukan ganda
- Titik akhir VPC antarmuka
- AWS layanan yang bekerja dengan IAM
- AWS Tanda Tangan Versi 4 untuk permintaan API
- Referensi kebijakan IAM JSON

- Layanan AWS Informasi yang disederhanakan untuk akses terprogram

Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARN)

Nama Sumber Daya Amazon (ARN) mengidentifikasi AWS sumber daya secara unik. Kami memerlukan ARN ketika Anda perlu menentukan sumber daya secara jelas di semua, seperti dalam kebijakan IAM AWS, tag Amazon Relational Database Service (Amazon RDS), dan panggilan API. Sementara ARN, seperti informasi pengenal lainnya, harus digunakan dan dibagikan dengan hati-hati, ARN tidak dianggap informasi rahasia, sensitif, atau rahasia.

Format ARN

Berikut ini adalah format umum untuk ARN. Format spesifik bergantung pada sumber daya. Untuk menggunakan ARN, ganti *`italicized`* teks dengan informasi khusus sumber daya. Ketahuilah bahwa ARN untuk beberapa sumber daya menghilangkan Wilayah, ID akun, atau Wilayah dan ID akun.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

Partisi tempat sumber daya berada. Partisi adalah sekelompok Wil AWS ayah. Setiap AWS akun dicakup ke satu partisi.

Berikut ini adalah partisi yang didukung:

- `aws`- Wil AWS ayah
- `aws-cn` - Wilayah Tiongkok
- `aws-us-gov` - AWS GovCloud (US) Wilayah

service

Namespace layanan yang mengidentifikasi AWS produk.

region

Kode Wilayah. Misalnya, `us-east-2` untuk US East (Ohio). Untuk daftar kode Wilayah, lihat T itik akhir regional di Referensi Umum AWS.

account-id

ID AWS akun yang memiliki sumber daya, tanpa tanda hubung. Misalnya, `123456789012`.

resource-type

Jenis sumber daya . Misalnya, `vpc` untuk cloud pribadi virtual (VPC).

resource-id

Pengidentifikasi sumber daya. Ini adalah nama sumber daya, ID sumber daya, atau jalur sumber daya. Beberapa pengidentifikasi sumber daya menyertakan sumber daya induk (sub-resource-type/parent - resource/sub -resource) atau kualifikasi seperti versi (resource-type:resource-name:qualifier).

Contoh

Pengguna IAM

arn:aws:iam: ::penggun/ *123456789012 john*

Topik SNS

arn: aws:sns::: *us-east-1 123456789012 example-sns-topic-name*

VPC

arn: aws:ec2: ::vpc/*us-east-1123456789012vpc-0e9801d129EXAMPLE*

Cari format ARN untuk sumber daya

Format ARN yang tepat tergantung pada jenis layanan dan sumber daya. Beberapa ARN sumber daya dapat menyertakan jalur, variabel, atau wildcard. Untuk mencari format ARN untuk AWS sumber daya tertentu, buka Referensi Otorisasi Layanan, buka halaman untuk layanan, dan navigasikan ke tabel jenis sumber daya.

Jalur di ARN

ARN Sumber Daya dapat menyertakan jalur. Misalnya, di Amazon S3, pengenal sumber daya adalah nama objek yang dapat menyertakan garis miring maju (/) untuk membentuk jalur. Demikian pula, nama pengguna dan nama grup IAM dapat menyertakan jalur. Hanya karakter alfanumerik dan karakter berikut yang diperbolehkan di jalur IAM: garis miring maju (/), plus (+), sama dengan (=), koma (), titik (,), at (.), garis bawah (@), dan tanda hubung (_). -

Menggunakan wildcard di jalur

Jalur dapat menyertakan karakter wildcard, yaitu tanda bintang (*). Beberapa elemen kebijakan mengizinkan wildcard sementara yang lain tidak. Anda dapat menggunakan wildcard untuk Sumber Daya atau NotResource elemen tetapi tidak untuk Prinsip ??? atau NotPrincipal elemen. Untuk informasi selengkapnya, lihat Referensi kebijakan IAM JSON.

Anda dapat menentukan `role/*` untuk berarti semua peran dalam akun 123456789012 seperti pada contoh berikut:

```
arn:aws:iam::123456789012:role/*
```

Anda juga dapat mengakhiri nama sumber daya dengan wildcard. Misalnya, Anda dapat menentukan **`service-*`** untuk berarti semua peran dimulai dengan `service` dan diakhiri dengan karakter yang berbeda seperti `service-role1` atau`service-test`:

```
arn:aws:iam::123456789012:role/service-*
```

Contoh berikut menunjukkan ARN untuk objek dalam bucket Amazon S3 di mana nama sumber daya menyertakan jalur. ARN `arn:aws:s3:::amzn-s3-demo-bucket/*` adalah untuk semua objek dalam bucket itu, terlepas dari awalan. ARN `arn:aws:s3:::amzn-s3-demo-bucket/Development/*` adalah untuk semua objek yang dibuat dalam **`/Development/`** awalan.

Anda juga dapat menggunakan karakter ? wildcard untuk menentukan satu karakter dalam ARN. Misalnya, Anda dapat menggunakan ARN berikut untuk semua folder yang dimulai dengan empat karakter dan berakhir **`-test`** di bucket S3 bernama amzn-s3-demo-bucket. Beberapa folder yang cocok dengan ini termasuk`1234-test`,`2024-test`, atau`100-test`.

```
arn:aws:s3:::amzn-s3-demo-bucket/????-test
```

Anda juga dapat menggunakan wildcard di berbagai bagian ARN, dibatasi oleh titik dua “”. **`:`** Dalam contoh berikut, dua wildcard digunakan untuk mencocokkan semua aplikasi dan sumber daya Amazon Q dalam aplikasi di semua wilayah untuk akun 123456789012:

```
arn:aws:qbusiness:*:123456789012:*
```

Demikian pula, contoh berikut cocok dengan semua VPC Amazon di semua wilayah untuk akun 123456789012:

```
arn:aws:ec2:*:123456789012:vpc/*
```

Contoh berikut cocok dengan semua volume Amazon EBS di semua wilayah untuk akun 123456789012:

```
arn:aws:ec2:*:123456789012:volume/*
```

Batasan penggunaan wildcard dalam ARN

Anda tidak dapat menggunakan wildcard di bagian ARN yang menentukan jenis sumber daya. Contoh ARN berikut dengan wildcard dalam tipe resource tidak valid:

```
arn:aws:lambda:us-east-2:123456789012:functi*:my-function <== not allowed
```

> **Note**
>
> Bila Anda menentukan ARN yang tidak lengkap (satu dengan kurang dari enam bidang standar) dalam kebijakan berbasis identitas, AWS secara otomatis melengkapi ARN

dengan menambahkan karakter wildcard (*) ke semua bidang yang hilang. Misalnya, menentukan `arn:aws:sqs` setara dengan`arn:aws:sqs:*:*:*`, yang memberikan akses ke semua sumber daya Amazon Amazon SQS di semua wilayah dan akun. Namun, kebijakan sesi yang diteruskan ke AWS STS AssumeRole, AssumeRoleWithWebIdentity, dan AssumeRoleWithSAML permintaan, tidak mendukung ARN yang tidak lengkap. Menggunakan ARN yang tidak lengkap dalam kebijakan sesi akan mengakibatkan `MalformedPolicyDocumentException` kesalahan.

Anda juga tidak dapat menggunakan wildcard di awalan ARN, atau memiliki wildcard di bagian partisi ARN.

```
arn:aws:redshift:us-east-1:123456789012:? <== not allowed
```

Pengidentifikasi IAM

IAM menggunakan beberapa pengenal berbeda untuk pengguna, grup IAM, peran, kebijakan, dan sertifikat server. Bagian ini menjelaskan pengidentifikasi dan kapan Anda menggunakan masing-masing pengidentifikasi.

Topik

- Nama dan jalur yang ramah
- ARN IAM
- Pengidentifikasi unik
- Memahami prefiks ID yang unik
- Mendapatkan pengidentifikasi unik

Nama dan jalur yang ramah

Saat Anda membuat pengguna, peran, grup pengguna, atau kebijakan, atau saat Anda mengunggah sertifikat server, Anda memberikan nama yang ramah. Contohnya termasuk Bob, TestApp 1, Pengembang ManageCredentialsPermissions,, atau ProdServerCert.

Jika Anda menggunakan IAM API atau AWS Command Line Interface (AWS CLI) untuk membuat sumber daya IAM, Anda dapat menambahkan jalur opsional. Anda dapat menggunakan satu jalur, atau sarang beberapa jalur sebagai struktur folder. Misalnya, Anda dapat menggunakan

jalur bersarang `/division_abc/subdivision_xyz/product_1234/engineering/` untuk mencocokkan struktur organisasi perusahaan Anda. Kemudian, Anda dapat membuat kebijakan yang memungkinkan semua pengguna di jalur tersebut mengakses API simulator kebijakan. Untuk melihat kebijakan ini, lihat IAM: Akses API simulator kebijakan berdasarkan jalur pengguna. Untuk informasi tentang cara menentukan nama yang mudah diingat, lihat dokumentasi API Pengguna. Untuk contoh-contoh tambahan tentang bagaimana Anda mungkin menggunakan jalur, lihat ARN IAM.

Saat Anda menggunakan CloudFormation untuk membuat sumber daya, Anda dapat menentukan jalur untuk pengguna, grup IAM, dan peran, serta kebijakan yang dikelola pelanggan.

Jika Anda memiliki pengguna dan grup pengguna di jalur yang sama, IAM tidak secara otomatis menempatkan pengguna dalam grup pengguna tersebut. Misalnya, Anda dapat membuat grup pengguna Pengembang dan menentukan jalur sebagai`/division_abc/subdivision_xyz/product_1234/engineering/`. Jika Anda membuat pengguna bernama Bob dan menambahkan jalur yang sama kepadanya, ini tidak secara otomatis menempatkan Bob di grup pengguna Pengembang. IAM tidak memberlakukan batasan apa pun antara pengguna atau grup IAM berdasarkan jalur mereka. Pengguna dengan jalur berbeda dapat menggunakan sumber daya yang sama jika mereka telah diberikan izin ke sumber daya tersebut. Jumlah dan ukuran sumber daya IAM dalam AWS akun terbatas. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota.

ARN IAM

Sebagian besar sumber daya memiliki nama ramah misalnya, nama pengguna `Bob` atau grup pengguna bernama`Developers`. Namun, bahasa kebijakan izin mengharuskan Anda untuk menentukan sumber daya menggunakan format Amazon Resource Name (ARN) berikut.

```
arn:partition:service:region:account:resource
```

Di mana:

- `partition`mengidentifikasi partisi untuk sumber daya. Untuk Wilayah AWS standar, partisinya adalah `aws`. Jika Anda memiliki sumber daya di partisi lain, maka partisi-nya adalah `aws-`*`partitionname`*. Contohnya, partisi untuk sumber daya di Wilayah Tiongkok (Beijing) adalah `aws-cn`. Anda tidak dapat mendelegasikan akses di antara akun di partisi yang berbeda.
- `service`mengidentifikasi AWS produk. Sumber daya IAM selalu digunakan`iam`.
- `region`mengidentifikasi wilayah sumber daya. Untuk sumber daya IAM, ini selalu dibiarkan kosong.
- `account`menentukan Akun AWS ID tanpa tanda hubung.

- `resource`mengidentifikasi sumber daya tertentu dengan nama.

Anda dapat menentukan IAM dan AWS STS ARN menggunakan sintaks berikut. Bagian Wilayah ARN kosong karena sumber daya IAM bersifat global.

Sintaks:

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
arn:aws:iam::aws:contextProvider/context-provider-name
```

Banyak dari contoh-contoh berikut mencakup jalur di bagian sumber daya ARN. Jalur tidak dapat dibuat atau dimanipulasi di Konsol Manajemen AWS. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS API, AWS CLI, atau Tools untuk Windows PowerShell.

Contoh:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/John
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

```
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/John
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Jane
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneMFA
arn:aws:iam::123456789012:u2f/user/John/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
arn:aws:iam::aws:contextProvider/IdentityCenter
```

Contoh berikut memberikan detail lebih lanjut untuk membantu Anda memahami format ARN untuk berbagai jenis IAM dan AWS STS sumber daya.

- Pengguna IAM di akun:

> **Note**
>
> Setiap nama pengguna IAM adalah unik. Nama pengguna tidak peka huruf besar/kecil untuk pengguna, seperti selama proses masuk, tetapi peka huruf besar/kecil saat Anda menggunakannya dalam kebijakan atau sebagai bagian dari ARN.

```
arn:aws:iam::123456789012:user/John
```

- Pengguna lain dengan jalur yang mencerminkan bagan organisasi:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
```

- Grup pengguna IAM:

```
arn:aws:iam::123456789012:group/Developers
```

- Grup pengguna IAM dengan jalur:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Peran IAM:

```
arn:aws:iam::123456789012:role/S3Access
```

- Peran yang terhubung dengan layanan:

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Peran layanan:

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Kebijakan terkelola:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Profil instans yang dapat dikaitkan dengan instans Amazon EC2:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Prinsip pengguna AWS STS federasi yang diidentifikasi di IAM sebagai “Paulo”:

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- Sesi aktif seseorang yang mengambil peran "Accounting-Role“, dengan nama sesi peran “Mary”:

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Merupakan sesi pemanggil sendiri saat digunakan sebagai sumber daya dalam panggilan API, seperti AWS STS SetContext API, yang beroperasi pada sesi pemanggilan:

```
arn:aws:sts::123456789012:self
```

- Perangkat autentikasi multi-faktor yang ditetapkan ke pengguna bernama Jorge:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Sertifikat server:

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Sertifikat server dengan jalur yang mencerminkan bagan organisasi:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Penyedia identitas (SAML dan OIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- Penyedia identitas OIDC dengan jalur yang mencerminkan URL penyedia identitas OIDC Amazon EKS:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

- Penyedia konteks ARN digunakan dengan propagasi identitas te AWS IAM Identity Center percaya, dari mana pernyataan konteks tepercaya dihasilkan:

```
arn:aws:iam::aws:contextProvider/IdentityCenter
```

ARN penting lainnya adalah ARN pengguna akar. Meskipun ini bukan sumber daya IAM, Anda harus familiar dengan format ARN ini. Hal ini sering digunakan dalam `Principal` elemen kebijakan berbasis sumber daya.

- Men Akun AWS ampilkan sebagai berikut:

```
arn:aws:iam::123456789012:root
```

Contoh berikut menunjukkan kebijakan yang dapat Anda tetapkan ke Richard untuk mengizinkannya mengelola kunci aksesnya. Perhatikan bahwa sumber daya adalah Richard, pengguna IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/
Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Saat menggunakan ARN untuk mengidentifikasi sumber daya dalam kebijakan IAM, Anda dapat menyertakan variabel kebijakan. Variabel kebijakan dapat mencakup placeholder untuk informasi waktu aktif (seperti nama pengguna) sebagai bagian dari ARN. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: Variabel dan tag

Menggunakan wildcard dan jalur di ARN

Anda dapat menggunakan wildcard di *`resource`* bagian ARN untuk menentukan beberapa pengguna atau grup IAM atau kebijakan. Misalnya, untuk menentukan semua pengguna yang bekerja pada product_1234, Anda menggunakan:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Jika Anda memiliki pengguna yang namanya dimulai dengan string`app_`, Anda dapat merujuk ke mereka semua dengan ARN berikut.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Untuk menentukan semua pengguna, grup IAM, atau kebijakan di Anda Akun AWS, gunakan wildcard setelah`user/`,`group/`, atau `policy/` bagian dari ARN, masing-masing.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Jika Anda menentukan ARN berikut untuk pengguna `arn:aws:iam::111122223333:user/*`, ini cocok dengan kedua contoh berikut.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Namun, jika Anda menentukan ARN berikut untuk penggunaa`rn:aws:iam::111122223333:user/division_abc*`, ini cocok dengan contoh kedua, tetapi bukan yang pertama.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Jane
```

Jangan gunakan wildcard di `user/`, `group/`, atau `policy/` bagian dari ARN. Misalnya, IAM tidak mengizinkan hal berikut:

```
arn:aws:iam::123456789012:u*
```

> **Note**
>
> Bila Anda menentukan ARN yang tidak lengkap (satu dengan kurang dari enam bidang standar) dalam kebijakan berbasis identitas, AWS secara otomatis melengkapi ARN dengan menambahkan karakter wildcard (*) ke semua bidang yang hilang. Misalnya, menentukan `arn:aws:sqs` setara dengan`arn:aws:sqs:*:*:*`, yang memberikan akses ke semua sumber daya Amazon Amazon SQS di semua wilayah dan akun. Namun,

kebijakan sesi yang diteruskan ke AWS STS AssumeRole, AssumeRoleWithWebIdentity, dan AssumeRoleWithSAML permintaan, tidak mendukung ARN yang tidak lengkap. Menggunakan ARN yang tidak lengkap dalam kebijakan sesi akan mengakibatkan `MalformedPolicyDocumentException` kesalahan.

Example Contoh penggunaan jalur dan ARN untuk grup pengguna berbasis proyek

Jalur tidak dapat dibuat atau dimanipulasi di Konsol Manajemen AWS. Untuk menggunakan jalur, Anda harus bekerja dengan sumber daya dengan menggunakan AWS API, AWS CLI, atau Alat untuk Windows PowerShell.

Dalam contoh ini, Jules dalam grup Marketing_Admin pengguna membuat grup pengguna berbasis proyek dalam jalur /marketing/. Jules menugaskan pengguna dari berbagai bagian perusahaan ke grup pengguna. Contoh ini menggambarkan bahwa jalur pengguna tidak terkait dengan grup pengguna yang pengguna ikuti.

Grup pemasaran memiliki produk baru yang akan mereka luncurkan, jadi Jules membuat grup pengguna baru di jalur /marketing/yang disebut. Widget_Launch Jules kemudian menetapkan kebijakan berikut kepada grup pengguna, yang memberi grup pengguna akses ke objek di bagian `example_bucket` yang ditetapkan untuk peluncuran khusus ini.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/
*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/
widget/*"}}
    }
```

```
  ]
}
```

Jules, kemudian menugaskan pengguna yang mengerjakan peluncuran ini ke grup pengguna. Ini termasuk Patricia dan Eli dari jalur /pemasaran/. Ini juga mencakup Chris dan Chloe dari jalur / pemasaran/, dan Alice serta Jim dari /legal/.

Pengidentifikasi unik

Saat IAM membuat pengguna, grup pengguna, peran, kebijakan, profil instans, atau sertifikat server, IAM menetapkan ID unik untuk setiap sumber daya. ID unik terlihat seperti ini:

`AIDAJQABLZS4A3QDU576Q`

Sebagian besar, Anda menggunakan nama ramah dan ARN saat bekerja dengan sumber daya IAM. Dengan begitu, Anda tidak perlu mengetahui ID unik untuk sumber daya tertentu. Namun, ID unik terkadang dapat berguna jika saat situasi tidak praktis untuk menggunakan nama yang ramah.

Salah satu contoh menggunakan kembali nama ramah di Anda Akun AWS. Di dalam akun Anda, nama ramah untuk pengguna, grup pengguna, peran, atau kebijakan harus unik. Misalnya, Anda dapat membuat pengguna IAM bernama`John`. Perusahaan Anda menggunakan Amazon S3 dan memiliki folder dengan bucket untuk setiap karyawan. Pengguna IAM `John` adalah anggota grup pengguna IAM yang diberi nama `User-S3-Access` dengan izin yang memungkinkan pengguna mengakses hanya ke folder mereka sendiri di bucket. Untuk contoh bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan pengguna IAM mengakses objek bucket mereka sendiri di S3 menggunakan nama pengguna yang ramah, lihat. Amazon S3: Memungkinkan pengguna IAM mengakses direktori home S3 mereka, secara terprogram dan di konsol

Misalkan karyawan bernama John meninggalkan perusahaan Anda dan Anda menghapus nama `John` pengguna IAM yang sesuai. Tetapi kemudian karyawan lain bernama John mulai, dan Anda membuat pengguna IAM baru bernama`John`. Anda menambahkan nama pengguna IAM baru `John` ke grup pengguna IAM yang ada. `User-S3-Access` Jika kebijakan yang terkait dengan grup pengguna menentukan nama pengguna IAM yang ramah`John`, kebijakan tersebut memungkinkan John baru mengakses informasi yang ditinggalkan oleh mantan John.

Secara umum, kami sarankan Anda menentukan ARN untuk sumber daya dalam kebijakan Anda, bukan ID uniknya. Namun, setiap pengguna IAM memiliki ID unik, bahkan jika Anda membuat pengguna IAM baru yang menggunakan kembali nama ramah yang Anda hapus sebelumnya. Dalam

contoh, pengguna IAM lama `John` dan pengguna IAM baru `John` memiliki ID unik yang berbeda. Anda dapat membuat kebijakan berbasis sumber daya yang memberikan akses berdasarkan ID unik dan bukan hanya berdasarkan nama pengguna. Melakukannya mengurangi kemungkinan bahwa Anda secara tidak sengaja dapat memberikan akses ke informasi yang seharusnya tidak dimiliki oleh seorang karyawan.

Contoh berikut menunjukkan cara menentukan ID unik dalam `Principal` elemen kebijakan berbasis sumber daya.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

Contoh berikut menunjukkan cara menentukan ID unik dalam `Condition` elemen kebijakan menggunakan kunci kondisi global `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Contoh lain di mana ID pengguna dapat berguna adalah jika Anda memelihara database Anda sendiri (atau penyimpanan lain) informasi pengguna atau peran IAM. ID unik dapat memberikan pengenal unik untuk setiap pengguna atau peran IAM yang Anda buat. Ini adalah kasus ketika Anda memiliki pengguna atau peran IAM yang menggunakan kembali nama, seperti pada contoh sebelumnya.

Memahami prefiks ID yang unik

IAM menggunakan awalan berikut untuk menunjukkan jenis sumber daya yang masing-masing ID unik berlaku. Awalan dapat bervariasi berdasarkan kapan mereka dibuat.

Awalan	Tipe sumber daya
ABIA	AWS STS token pembawa layanan
ACCA	Context-specific kredensi
AGPA	Grup pengguna
AIDA	Pengguna IAM
AIPA	Profil instans Amazon EC2
AKIA	Access key
ANPA	Kebijakan terkelola
ANVA	Versi dalam kebijakan terkelola
APKA	Kunci publik
AROA	Peran
ASCA	Sertifikat
ASIA	ID access key sementara (AWS STS) menggunakan prefiks ini, tetapi hanya unik dalam kombinasi dengan secret access key dan token sesi.

Mendapatkan pengidentifikasi unik

ID unik untuk sumber daya IAM tidak tersedia di konsol IAM. Untuk mendapatkan ID unik, Anda dapat menggunakan AWS CLI perintah berikut atau panggilan API IAM.

AWS CLI:

- dapatkan identitas penelepon
- dapatkan kelompok
- dapatkan-peran

- dapatkan pengguna
- mendapatkan kebijakan
- get-instance-profile
- dapatkan-server-sertifikat

API IAM:

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM dan AWS STS kuota

AWS Identity and Access Management (IAM) dan AWS Security Token Service (STS) memiliki kuota yang membatasi ukuran objek. Hal ini memengaruhi cara Anda menamai sebuah objek, jumlah objek yang dapat Anda buat, dan jumlah karakter yang dapat Anda gunakan saat Anda melewati sebuah objek.

> **Note**
>
> Untuk mendapatkan informasi tingkat akun tentang penggunaan dan kuota IAM, gunakan operasi GetAccountSummary API atau perintah get-account-summary. https://docs.aws.amazon.com/cli/latest/reference/iam/get-account-summary.html AWS CLI

Persyaratan nama IAM

Nama IAM memiliki persyaratan dan pembatasan berikut:

- Dokumen kebijakan hanya dapat berisi karakter Unicode berikut: tab horizontal (U+0009), umpan baris (U+000A), pengembalian pengangkutan (U+000D), dan karakter dalam rentang U+0020 hingga U+00FF.
- Nama pengguna, grup, peran, kebijakan, profil instans, sertifikat server, dan jalur harus alfanumerik, termasuk karakter umum berikut: plus (+), sama dengan (=), koma (,), titik (.), at (@), garis bawah (_), dan tanda hubung (-). Nama jalur harus dimulai dan diakhiri dengan garis miring ke depan (/).
- Nama pengguna, grup, peran, dan profil instans harus unik dalam akun. Mereka tidak dibedakan berdasarkan kasus, misalnya, Anda tidak dapat membuat grup bernama keduanya **`ADMINS`** dan**`admins`**.
- Nilai ID eksternal yang digunakan pihak ketiga untuk mengambil peran harus memiliki minimal 2 karakter dan maksimal 1.224 karakter. Nilai harus berupa alfanumerik tanpa spasi. Nilai dapat mencakup simbol berikut: plus (+), setara (=), koma (,), titik (.), a keong (@), titik dua (:), garis miring (/), dan tanda hubung (-). Untuk informasi selengkapnya tentang ID eksternal, lihat Akses ke Akun AWS dimiliki oleh pihak ketiga.
- Nama kebijakan untuk kebijakan sebaris harus unik untuk pengguna, grup, atau peran yang disematkan. Nama-nama dapat berisi karakter Basic Latin (ASCII) kecuali untuk karakter cadangan berikut: garis miring mundur (\), garis miring maju (/), tanda bintang (*), tanda tanya (?) , dan ruang putih. Karakter-karakter ini dicadangkan menurut RFC 3986, bagian 2.2.
- Kata sandi pengguna (profil masuk) dapat mengandung karakter Latin Basic (ASCII).
- Akun AWS Alias ID harus unik di seluruh AWS produk, dan harus alfanumerik mengikuti konvensi penamaan DNS. Alias harus huruf kecil, tidak boleh dimulai atau diakhiri dengan tanda hubung, tidak dapat berisi dua tanda hubung berturut-turut, dan tidak boleh berupa angka 12 digit.

Untuk daftar karakter Basic Latin (ASCII), lihat Tabel Kode Library of Congress Basic Latin (ASCII).

Kuota objek IAM

Kuota, juga disebut sebagai batas di AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di Anda Akun AWS. Gunakan Kuota Layanan untuk mengelola kuota IAM Anda.

Untuk daftar titik akhir layanan IAM dan kuota layanan, lihat AWS Identity and Access Management titik akhir dan kuota di. Referensi Umum AWS

Untuk meminta kenaikan kuota

1. Ikuti prosedur masuk yang sesuai dengan jenis pengguna Anda seperti yang dijelaskan dalam topik Cara masuk AWS di Panduan AWS Sign-In Pengguna untuk masuk ke Konsol Manajemen AWS.
2. Buka Konsol Service Quotas.
3. Di panel navigasi, pilih Layanan AWS .
4. Di bilah navigasi, pilih Wilayah AS Timur (N. Virginia). Lalu cari **`IAM`**.
5. Pilih AWS Identity and Access Management (IAM), pilih kuota, dan ikuti petunjuk untuk meminta kenaikan kuota.

Untuk informasi lebih lanjut, lihat Meminta Kenaikan Kuota dalam Panduan Pengguna Kuota Layanan.

Untuk melihat contoh cara meminta peningkatan kuota IAM menggunakan konsol Kuota Layanan, tonton video berikut.

Minta peningkatan kuota IAM menggunakan konsol Kuota Layanan.

Anda dapat meminta peningkatan kuota default untuk kuota IAM yang dapat disesuaikan. Permintaan hingga maximum quota secara otomatis disetujui dan diselesaikan dalam beberapa menit.

Tabel berikut mencantumkan sumber daya yang kenaikan kuota dapat disetujui secara otomatis.

Sumber daya	Kuota default	Kuota maksimum
Kebijakan yang dikelola pelanggan dalam akun	1500	10000
Grup kunci per akun	300	500
Profil instans per akun	1000	10000
Kebijakan terkelola per peran	20	25
Kebijakan terkelola per pengguna	10	20
Kebijakan terkelola per grup	10	10

Sumber daya	Kuota default	Kuota maksimum
Panjang kebijakan kepercaya an peran	2048 karakter	8192 karakter
Tabel Per Akun	1000	10000
Sertifikat server per akun	20	20
OpenId menghubungkan penyedia per akun	100	700

Kuota Penganalisis Akses IAM

Untuk daftar titik akhir layanan Penganalisis Akses IAM dan kuota layanan, lihat titik akhir dan kuota Penganalisis Akses IAM di. Referensi Umum AWS

Kuota IAM Roles Anywhere

Untuk daftar titik akhir layanan dan kuota layanan IAM Roles Anywhere, lihat titik akhir dan kuota AWS Identity and Access Management Roles Anywhere di. Referensi Umum AWS

Kuota permintaan STS

AWS Security Token Service (AWS STS) memberlakukan kuota permintaan berikut.

Untuk AWS STS permintaan yang dibuat menggunakan AWS kredenSIAL, kuota permintaan default adalah 600 permintaan per detik, per akun, per Wilayah. AWS STS Operasi berikut berbagi kuota ini:

- AssumeRole
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

> **Note**
>
> Permintaan AWS STS oleh AWS prinsipal layanan, seperti yang digunakan untuk mengambil peran untuk digunakan dengan AWS layanan, tidak menggunakan kuota permintaan STS per detik di akun Anda.

Misalnya, jika an Akun AWS membuat 100 GetCallerIdentity permintaan per detik dan 100 AssumeRole panggilan per detik di wilayah yang sama, akun tersebut menghabiskan 200 dari 600 permintaan STS yang tersedia per detik untuk wilayah tersebut.

Untuk AssumeRole permintaan lintas akun, hanya akun yang membuat AssumeRole permintaan yang memengaruhi kuota STS. Akun target tidak memiliki kuota yang dikonsumsi.

Untuk meminta peningkatan kuota permintaan STS, silakan buka tiket dengan AWS dukungan.

> **Note**
>
> Dengan perubahan yang akan datang pada titik akhir AWS STS global (`https://sts.amazonaws.com`), permintaan ke titik akhir global tidak akan berbagi kuota permintaan per detik (RPS) dengan titik akhir AWS STS Regional di Wilayah yang di aktifkan secara default. Ketika permintaan ke titik akhir AWS STS global berasal dari satu Wilayah, permintaan tersebut akan dihitung terhadap kuota RPS titik akhir global. Namun, ketika permintaan datang dari beberapa Wilayah, setiap Wilayah tambahan akan menerima kuota RPS independennya sendiri. Untuk informasi selengkapnya tentang perubahan titik akhir AWS STS global, lihatAWS STS perubahan titik akhir global.

Batas karakter IAM dan STS

Berikut ini adalah jumlah karakter maksimum dan batas ukuran untuk IAM dan AWS STS. Anda tidak dapat meminta kenaikan untuk batas berikut.

Deskripsi	Kuota
Alias untuk Akun AWS ID	3–63 karakter
Untuk kebijakan inline	Anda dapat menambahkan kebijakan inline sebanyak yang Anda inginkan kepada

Deskripsi	Kuota
	pengguna, peran, atau grup IAM. Tetapi ukuran kebijakan agregat total (ukuran jumlah semua kebijakan sebaris) per entitas tidak dapat melebihi batas berikut: • Ukuran kebijakan pengguna tidak boleh melebihi 2.048 karakter. • Ukuran kebijakan peran tidak boleh melebihi 10.240 karakter. • Ukuran kebijakan grup tidak boleh melebihi 5.120 karakter. **Note** IAM tidak menghitung spasi putih saat menghitung ukuran kebijakan terhadap batas ini.
Untuk kebijakan yang dikelola pelanggan	• Ukuran setiap kebijakan terkelola tidak boleh melebihi 6.144 karakter. **Note** IAM tidak menghitung spasi putih saat menghitung ukuran kebijakan terhadap batas ini.
Nama grup	128 karakter
Nama profil instans	128 karakter
Kata sandi untuk profil masuk	1–128 karakter
Jalur	512 karakter

Deskripsi	Kuota
Nama kebijakan	128 karakter
Nama peran	64 karakter **Important** Jika Anda bermaksud menggunakan peran dengan fitur Ber alih Peran di Konsol Manajemen AWS, maka gabungan `Path` dan tidak `RoleName` boleh melebihi 64 karakter.
Durasi sesi peran	12 jam Saat Anda mengambil peran dari API AWS CLI or, Anda dapat menggunakan parameter `duration-seconds` CLI atau parameter `DurationSeconds` API untuk meminta sesi peran yang lebih lama. Anda dapat menentukan nilai dari 900 detik (15 menit) hingga pengaturan durasi sesi maksimum untuk peran tersebut, yang dapat berkisar antara 1–12 jam. Jika Anda tidak menentuka n nilai untuk parameter `DurationSeconds` tersebut, kredensial keamanan Anda berlaku selama satu jam. Pengguna IAM yang beralih peran di konsol diberikan durasi sesi maksimum, atau sisa waktu dalam sesi pengguna, mana saja yang kurang. Pengatura n durasi sesi maksimum tidak membatasi sesi yang diasumsikan oleh AWS layanan. Untuk mempelajari cara melihat nilai maksimum untuk peran Anda, lihat Memperbarui durasi sesi maksimum untuk peran.

Deskripsi	Kuota
Nama sesi peran	64 karakter
Peran kebijakan sesi	• Ukuran dokumen kebijakan JSON yang diteruskan dan gabungan semua karakter ARN kebijakan terkelola yang dilewati tidak boleh melebihi 2.048 karakter. • Anda dapat meneruskan maksimum 10 ARN kebijakan terkelola saat membuat sesi. • Anda hanya dapat meneruskan satu dokumen kebijakan JSON saat Anda membuat sesi sementara secara terprogram untuk peran atau prinsipal pengguna AWS STS federasi. • Selain itu, kebijakan sesi dan tag sesi menambah ukuran token sesi, yang memiliki batas terpisah. Elem `SessionTokenUtilization` en respons menunjukkan berapa persentase dari batas ukuran yang digunakan token. Elem `PackedPolicySize` en respons tidak digunakan lagi. Gunakan `SessionTokenUtilization` sebagai gantinya. • Kami menyarankan Anda melewati kebijakan sesi menggunakan AWS API AWS CLI or. Konsol Manajemen AWS Mungkin meningkatkan ukuran token sesi dengan menambahkan informasi sesi.

Deskripsi	Kuota
Peran tanda sesi	• Tag sesi harus memenuhi batas kunci tag 128 karakter dan batas nilai tag 256 karakter. • Anda dapat meneruskan hingga 50 tanda sesi. • Kebijakan sesi dan tag sesi menambah ukuran token sesi, yang memiliki batas terpisah. Anda dapat meneruskan tag sesi menggunakan AWS API AWS CLI or. Elem `SessionTokenUtilization` en respons menunjukkan berapa persentase dari batas ukuran yang digunakan token. Elem `PackedPolicySize` en respons tidak digunakan lagi. Gunakan `SessionTokenUtilization` sebagai gantinya.
Respons otentikasi SAML base64 dikodekan	100.000 karakter Batas karakter ini berlaku untuk operasi `assume-role-with-saml` CLI atau `AssumeRoleWithSAML` API.
Tombol tanda	128 karakter Batas karakter ini berlaku untuk tag pada sumber daya IAM dan tag ??? sesi.
Nilai tanda	256 karakter Batas karakter ini berlaku untuk tag pada sumber daya IAM dan tag ??? sesi. Nilai tag dapat kosong yang berarti nilai tag dapat memiliki panjang 0 karakter.

Deskripsi	Kuota
ID unik yang dibuat oleh IAM	128 karakter. Sebagai contoh: • ID Pengguna yang dimulai dengan `AIDA` • ID Grup yang dimulai dengan `AGPA` • ID Peran yang dimulai dengan `AROA` • ID kebijakan terkelola yang dimulai dengan `ANPA` • ID sertifikat server yang dimulai dengan `ASCA` **Note** Ini tidak dimaksudkan untuk menjadi daftar lengkap, juga bukan jaminan bahwa ID dari jenis tertentu hanya dimulai dengan kombinasi huruf yang ditentukan.
Nama pengguna	64 karakter

Dukungan titik akhir tumpukan ganda

Dual-stack endpoint memungkinkan klien untuk berkomunikasi dengan AWS layanan menggunakan salah satu atau IPv4 alamat. IPv6 Baik IAM dan AWS STS menyediakan dukungan endpoint dual-stack. Untuk informasi selengkapnya tentang cara mengonfigurasi VPC Anda untuk mode dual-stack, lihat dukungan IPv6 untuk VPC Anda di Panduan Pengguna Amazon VPC.

Jika Anda menggunakan kebijakan IAM yang menyertakan `aws:sourceIp` atau kunci `aws:vpcSourceIp` kondisi, Anda perlu memperbarui kebijakan ini untuk mendukung IPv6 klien. Untuk informasi selengkapnya tentang IPv6 pengalamatan untuk Anda VPCs, lihat pengalamatan IP untuk subnet Anda VPCs dan subnet di Panduan Pengguna Amazon VPC.

Dukungan titik akhir tumpukan ganda IAM

IAM menyediakan endpoint publik dual-stack yang mendukung keduanya dan klien. IPv4 IPv6 Titik akhir publik dual-stack IAM juga dapat diakses secara pribadi dari virtual private cloud (VPC) Anda menggunakan. AWS PrivateLink Untuk informasi selengkapnya tentang membuat titik akhir VPC antarmuka pribadi untuk IAM, lihat. Buat titik akhir VPC untuk IAM

Titik akhir publik dual-stack IAM adalah. `https://iam.global.api.aws`

Titik akhir publik IAM di `https://iam.amazonaws.com`, tidak seperti titik akhir publik tumpukan ganda, hanya mendukung klien. IPv4 Saat diakses secara pribadi dari VPC Anda AWS PrivateLink menggunakan, titik akhir publik IAM dapat mendukung keduanya dan klien. IPv4 IPv6

AWS STS dukungan titik akhir dual-stack

AWS STS menyediakan titik akhir regional dual-stack yang mendukung keduanya IPv4 dan klien. IPv6 Titik akhir regional AWS STS dual-stack juga dapat diakses secara pribadi dari virtual private cloud (VPC) Anda menggunakan. AWS PrivateLink Untuk informasi selengkapnya tentang membuat titik akhir VPC antarmuka pribadi, lihat. AWS STSBuat titik akhir VPC untuk AWS STS

Tabel berikut menunjukkan titik akhir regional AWS STS dual-stack menurut partisi:

Partition	URL titik akhir tumpukan ganda
AWS Daerah Komersil	Titik akhir reguler: `https://sts.` *`<region>`*`.api.aws` Titik akhir FIPS: `https://sts-fips.` *`<region>`*`.api.aws`
AWS GovCloud (US) Daerah	Titik akhir reguler: `https://sts.` *`<region>`*`.api.aws`
Wilayah Tiongkok	Titik akhir reguler: `https://sts.` *`<region>`*`.api.amazonwebservices.com.cn`

> **Note**
>
> Titik akhir FIPS hanya berlaku untuk wilayah AS dan Kanada.

Batasan

Batasan berikut berlaku untuk dukungan titik akhir AWS STS dual-stack:

- IPv6 klien tidak didukung pada titik akhir `https://sts.amazonaws.com` global. Anda harus menggunakan titik akhir regional dual-stack untuk IPv6 dukungan klien.
- IPv6-hanya titik akhir AWS STS VPC yang tidak didukung. Titik akhir VPC dapat dikonfigurasi untuk IPv4 atau konektivitas dual-stack.

Titik akhir VPC antarmuka

Jika Anda menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk meng-host AWS sumber daya Anda, Anda dapat membuat koneksi pribadi antara VPC dan AWS Identity and Access Management (IAM) atau AWS Security Token Service ().AWS STS Anda dapat menggunakan koneksi ini untuk mengaktifkan IAM atau AWS STS untuk berkomunikasi dengan sumber daya Anda di VPC Anda tanpa melalui internet publik.

Amazon VPC adalah AWS layanan yang dapat Anda gunakan untuk meluncurkan AWS sumber daya di jaringan virtual yang Anda tentukan. Dengan VPC, Anda memiliki kendali terhadap pengaturan jaringan, seperti rentang alamat IP, subnet, tabel rute, dan gateway jaringan. Untuk menghubungkan VPC Anda ke IAM atau AWS STS, Anda menentukan titik akhir VPC antarmuka untuk setiap layanan. Titik akhir menyediakan konektivitas yang andal dan dapat diskalakan ke IAM atau AWS STS tanpa memerlukan gateway internet, instance terjemahan alamat jaringan (NAT), atau koneksi VPN. Untuk informasi selengkapnya, lihat Apa yang dimaksud dengan Amazon VPC? dalam Panduan Pengguna Amazon VPC.

Titik akhir antarmuka VPC didukung oleh AWS teknologi AWS PrivateLink yang memungkinkan komunikasi pribadi antar AWS layanan menggunakan antarmuka jaringan elastis dengan alamat IP pribadi. Untuk informasi selengkapnya, lihat AWS PrivateLink untuk AWS Layanan.

Informasi berikut adalah untuk pengguna Amazon VPC. Untuk informasi selengkapnya, lihat Memulai dengan Amazon VPC dalam Panduan Pengguna Amazon VPC.

Topik

- Ketersediaan titik akhir VPC
- Buat titik akhir VPC untuk IAM
- Buat titik akhir VPC untuk AWS STS
- Buat titik akhir VPC untuk AWS STS Penemuan OIDC
- Kontrol akses ke AWS STS dengan kebijakan titik akhir VPC

- Gunakan tombol kondisi VPC untuk mengontrol akses federasi

Ketersediaan titik akhir VPC

> **Important**
>
> Antarmuka titik akhir VPC untuk IAM hanya dapat dibuat di Wilayah tempat bidang kontrol IAM berada. Jika VPC Anda berada di Wilayah yang berbeda dari Wilayah bidang kontrol IAM, Anda harus menggunakan AWS Transit Gateway untuk mengizinkan akses ke titik akhir VPC antarmuka IAM dari Wilayah lain. Untuk informasi selengkapnya, lihat Buat titik akhir VPC untuk IAM.

IAM saat ini mendukung titik akhir VPC di Wilayah berikut:

- AS Timur (Virginia Utara)
- Tiongkok (Beijing)
- AWS GovCloud (US-West)

AWS STS saat ini mendukung titik akhir VPC di Wilayah berikut:

- Timur AS (N. Virginia)
- AS Timur (Ohio)
- AS Barat (California Utara)
- AS Barat (Oregon)
- Afrika (Cape Town)
- Asia Pasifik (Hong Kong)
- Asia Pasifik (Hyderabad)
- Asia Pasifik (Jakarta)
- Asia Pasifik (Melbourne)
- Asia Pasifik (Mumbai)
- Asia Pasifik (Osaka)
- Asia Pasifik (Seoul)
- Asia Pasifik (Singapura)

- Asia Pasifik (Sydney)
- Asia Pasifik (Tokyo)
- (Canada (Central)
- Kanada Barat (Calgary)
- China (Beijing)
- Tiongkok (Ningxia)
- Eropa (Frankfurt)
- Eropa (Irlandia)
- Eropa (London)
- Eropa (Milan)
- Eropa (Paris)
- Eropa (Spanyol)
- Eropa (Stockholm)
- Eropa (Zürich)
- Israel (Tel Aviv)
- Timur Tengah (Bahrain)
- Timur Tengah (UEA)
- Amerika Selatan (Sao Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Buat titik akhir VPC untuk IAM

Untuk mulai menggunakan IAM dengan VPC Anda, buat antarmuka VPC endpoint untuk IAM. Untuk informasi selengkapnya, lihat Mengakses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan Pengguna Amazon VPC.

Titik akhir VPC antarmuka untuk IAM hanya dapat dibuat di Wilayah tempat bidang kontrol IAM berada. Secara komersial Wilayah AWS, pesawat kontrol IAM terletak di Wilayah AS Timur (Virginia N.) (us-east-1). Nama layanan titik akhir VPC AWS PrivateLink antarmuka untuk IAM adalah. `com.amazonaws.iam` Untuk daftar titik akhir VPC Wilayah AWS yang mendukung IAM, lihat. Ketersediaan titik akhir VPC

Jika VPC Anda berada di Wilayah yang berbeda dari Wilayah bidang kontrol IAM, Anda harus menggunakan AWS Transit Gateway untuk mengizinkan akses ke titik akhir VPC antarmuka IAM dari Wilayah lain.

Untuk mengakses antarmuka IAM VPC endpoint dari VPC di Wilayah yang berbeda menggunakan AWS Transit Gateway

1. Buat gateway transit, atau gunakan gateway transit yang ada untuk menghubungkan cloud pribadi virtual Anda (VPCs). Gateway transit diperlukan untuk setiap Wilayah. Untuk informasi selengkapnya, lihat Membuat gateway transit di AWS Transit Gateway Panduan.
2. Buat lampiran VPC gateway transit untuk menghubungkan setiap VPC ke gateway transit. Untuk informasi selengkapnya, lihat Membuat lampiran gateway transit ke VPC di Panduan.AWS Transit Gateway
3. Buat lampiran peering VPC gateway transit untuk merutekan lalu lintas antara peered. VPCs Untuk informasi selengkapnya, lihat Membuat lampiran peering di AWS Transit Gateway Panduan.

> **Note**
>
> Koneksi peering VPC juga dapat merutekan lalu lintas antara peered VPCs, tetapi metode ini tidak berskala baik dengan sejumlah besar. VPCs Alih-alih mengintip VPC, kami merekomendasikan AWS Transit Gateway lampiran peering yang meningkatkan VPC dan manajemen jaringan lokal melalui hub pusat yang dapat diskalakan. Untuk informasi selengkapnya tentang koneksi peering VPC, lihat Bekerja dengan koneksi peering VPC di Panduan Peering VPC Amazon.

Buat titik akhir VPC untuk AWS STS

Untuk mulai menggunakan AWS STS dengan VPC Anda, buat antarmuka titik akhir VPC untuk. AWS STS Untuk informasi selengkapnya, lihat Meng akses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan Pengguna Amazon VPC.

Setelah membuat titik akhir VPC, Anda harus menggunakan titik akhir regional yang cocok untuk mengirim AWS STS permintaan Anda. AWS STS merekomendasikan agar Anda menggunakan kedua `setEndpoint` metode `setRegion` dan untuk melakukan panggilan ke titik akhir Regional. Anda hanya dapat menggunakan metode `setRegion` untuk Wilayah yang diaktifkan secara

manual, seperti Asia Pacific (Hong Kong). Dalam kasus ini, panggilan akan diarahkan ke titik akhir Regional STS. Untuk mempelajari cara mengaktifkan sebuah Wilayah secara manual, lihat Mengelola Wilayah AWS dalam Referensi Umum AWS. Jika Anda hanya menggunakan metode `setRegion` untuk Wilayah yang diaktifkan secara default, panggilan diarahkan ke titik akhir global `https://sts.amazonaws.com` .

Saat Anda menggunakan titik akhir regional, AWS STS memanggil AWS layanan lain menggunakan titik akhir publik atau titik akhir VPC antarmuka pribadi, mana saja yang digunakan. Misalnya, asumsikan bahwa Anda telah membuat titik akhir VPC antarmuka untuk AWS STS dan telah meminta kredentif sementara AWS STS dari sumber daya yang terletak di VPC Anda. Dalam hal itu, kredensial ini mulai mengalir melalui antarmuka VPC endpoint secara default. Untuk informasi selengkapnya tentang membuat permintaan Regional menggunakan AWS STS, lihatKelola AWS STS dalam sebuah Wilayah AWS.

Buat titik akhir VPC untuk AWS STS Penemuan OIDC

AWS Security Token Service (AWS STS) Penemuan OIDC adalah sepasang titik akhir publik yang melayani metadata dan kunci verifikasi yang dibutuhkan pihak yang bergantung untuk memvalidasi Token Web JSON (JWT) yang dikeluarkan oleh federasi identitas keluar.

Saat Anda mengaktifkan federasi identitas keluar, AWS STS menerbitkan URL penerbit khusus akun. URL ini menampung dua titik akhir penemuan OpenID Connect (OIDC). Pihak yang mengandalkan mengambil titik akhir ini untuk mendapatkan metadata dan kunci publik yang mereka butuhkan untuk memverifikasi JWT yang mengeluarkan: GetWebIdentityToken

- `{issuer_url}/.well-known/openid-configuration`— Dokumen penemuan OIDC
- `{issuer_url}/.well-known/jwks.json`— Set Kunci Web JSON (JWKS) yang berisi kunci verifikasi tanda tangan token

Secara default, mencapai titik akhir ini dari VPC Anda memerlukan gateway internet, perangkat NAT, koneksi VPN, atau koneksi. Untuk menjaga lalu lintas ini di AWS jaringan sebagai gantinya, buat titik akhir VPC antarmuka untuk titik akhir penemuan STS OIDC.

> **Note**
>
> Titik akhir penemuan STS OIDC hanya melayani dua titik akhir yang tercantum di atas. Itu tidak melayani operasi AWS STS API apa pun. Untuk memang `GetWebIdentityToken`

gilAssumeRole,, atau AWS STS operasi lainnya secara pribadi, gunakan layanan `com.amazonaws.region.sts` titik akhir. Lihat Buat titik akhir VPC untuk AWS STS.

Memilih layanan endpoint yang tepat

Federasi identitas keluar menggunakan dua jalur jaringan yang berbeda. Sebagian besar beban kerja yang mencetak dan memverifikasi token memerlukan kedua layanan titik akhir.

Tugas	Layanan titik akhir	Autentikasi
Mint untuk JWT () `sts:GetWebIdentityToken`	`com.amazonaws.region.sts`	AWS SigV4
Membaca `/.well-known/openid-configuration`	`com.amazonaws.region.sts-oidc`	Tidak ada (tidak diautentikasi)
Membaca `/.well-known/jwks.json`	`com.amazonaws.region.sts-oidc`	Tidak ada (tidak diautentikasi)

Anda tidak memerlukan titik akhir penemuan STS OIDC jika satu-satunya pihak yang memverifikasi token Anda adalah layanan eksternal di luar. AWS Layanan itu mengambil URL penerbit Anda melalui internet publik, dan lalu lintasnya tidak masuk ke VPC Anda.

Pertimbangan-pertimbangan

Pertimbangkan hal berikut sebelum Anda membuat titik akhir VPC antarmuka untuk penemuan STS OIDC.

- Kebijakan titik akhir VPC tidak didukung. Kedua API penemuan melayani data yang tersedia untuk umum, dan titik akhir penemuan STS OIDC tidak menerapkan kebijakan titik akhir VPC. Gunakan grup keamanan dan penempatan subnet untuk mengontrol sumber daya mana yang dapat mencapai titik akhir.
- Hanya dua jalur yang dilayani. Titik akhir menerima `GET` permintaan HTTPS ke `/.well-known/openid-configuration` dan `/.well-known/jwks.json` hanya. Semua jalur dan metode lain mengembalikan kesalahan.

Membuat titik akhir VPC antarmuka untuk penemuan STS OIDC

Anda dapat membuat titik akhir VPC untuk titik akhir penemuan STS OIDC menggunakan konsol Amazon Virtual Private Cloud atau. AWS Command Line Interface Untuk informasi selengkapnya, lihat Meng akses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan Pengguna Amazon VPC.

Buat titik akhir VPC untuk penemuan STS OIDC menggunakan nama layanan berikut:

- `com.amazonaws.region.sts-oidc`

Jika Anda mengaktifkan DNS pribadi untuk titik akhir, Anda dapat mengirim permintaan ke `sts-oidc.region.amazonaws.com` dan permintaan tersebut akan diselesaikan ke antarmuka jaringan titik akhir dalam VPC Anda.

Mengirim permintaan melalui titik akhir

Ketika pihak yang mengandalkan di VPC Anda perlu memverifikasi JWT, ia mengambil dokumen penemuan OIDC dan JWKS dari URL penerbit. Nama host penerbit (misalnya,`uuid.tokens.sts.global.api.aws`) menggunakan domain yang berbeda dari endpoint (`sts-oidc.region.amazonaws.com`), sehingga tidak menyelesaikan ke titik akhir VPC secara default.

Untuk merutekan permintaan nama host penerbit melalui titik akhir, buat zona host pribadi Amazon Route 53 untuk nama host penerbit Anda dan tambahkan catatan alias yang menunjuk ke titik akhir VPC. Setelah Anda membuat zona yang dihosting, sumber daya di VPC menyelesaikan nama host penerbit ke antarmuka jaringan titik akhir. Pustaka OIDC standar kemudian dapat mengambil dokumen penemuan tanpa perubahan kode. Untuk informasi selengkapnya tentang membuat sebuah zona yang di-hosting privat, lihat Bekerja dengan Zona yang di-hosting privat di Panduan Developer Amazon Route 53.

> **Note**
>
> Layanan titik akhir bersifat regional, tetapi titik akhir tunggal melayani permintaan penemuan untuk setiap UUID penerbit di partisi. Buat titik akhir di Wilayah tempat verifikasi Anda berjalan; Anda tidak memerlukan satu per akun penerbit.

Kontrol akses ke AWS STS dengan kebijakan titik akhir VPC

Saat Anda membuat titik akhir VPC antarmuka untuk AWS Security Token Service (AWS STS), Anda dapat melampirkan kebijakan titik akhir. Kebijakan mengontrol prinsipal mana yang dapat menggunakan titik akhir dan AWS STS tindakan apa yang dapat mereka lakukan. Jika Anda tidak melampirkan kebijakan, titik akhir menggunakan kebijakan default yang memungkinkan akses tak terbatas ke semua AWS STS tindakan untuk semua prinsipal.

Kebijakan titik akhir VPC tidak memberikan izin sendiri. Mereka bertindak sebagai batas tambahan yang bekerja bersama kebijakan lainnya. Kebijakan titik akhir dan kebijakan pemanggil yang berlaku harus mengizinkan permintaan agar berhasil.

Untuk informasi selengkapnya tentang kebijakan titik akhir VPC, lihat Meng ontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan Pengguna Amazon VPC.

Topik

- Kebijakan VPC endpoint default
- Pertimbangan penting untuk AWS STS Kebijakan VPC endpoint
- Kunci kondisi tersedia untuk AWS STS Kebijakan VPC endpoint
- Contoh: Izinkan semua AWS STS Tindakan untuk organisasi Anda
- Contoh: Izinkan semua AWS STS tindakan untuk akun tertentu
- Contoh: Batasi ke spesifik AWS STS tindakan
- Contoh: Menolak akses non-organisasi sambil mengizinkan federasi

Kebijakan VPC endpoint default

Jika Anda tidak melampirkan kebijakan kustom saat membuat titik akhir, AWS lampirkan kebijakan default berikut. Kebijakan ini memungkinkan akses tak terbatas ke titik akhir.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
```

```
            "Resource": "*"
        }
    ]
}
```

Untuk membatasi akses ke titik akhir, lampirkan kebijakan titik akhir khusus.

Pertimbangan penting untuk AWS STS Kebijakan VPC endpoint

AWS STS menangani permintaan dari dua jenis penelepon yang berbeda secara fundamental. Kebijakan titik akhir VPC Anda harus memperhitungkan kedua jenis untuk menghindari pemblokiran permintaan yang sah secara tidak sengaja.

Prinsipal yang diautentikasi AWS

Pengguna IAM dan peran IAM yang menandatangani permintaan dengan Sig AWS nature Version 4 (SIGv4). Penelepon ini memiliki kunci kondisi standar seperti`aws:PrincipalOrgID`,`aws:PrincipalAccount`, dan `aws:PrincipalArn` tersedia dalam konteks permintaan.

Penelepon federasi

Prinsip SAML 2.0 dan OpenID Connect (OIDC) yang memanggil atau. `AssumeRoleWithSAML` `AssumeRoleWithWebIdentity` Pemanggil ini mengotentikasi dengan pernyataan SAML atau Token Web JSON (JWT), bukan tanda tangan SigV4. Karena mereka tidak memiliki AWS identitas pada saat permintaan, konteks permintaan tidak menyertakan kunci kondisi berbasis utama seperti`aws:PrincipalOrgID`,`aws:PrincipalAccount`, dan`aws:PrincipalArn`.

> **Important**
>
> Jika kebijakan titik akhir VPC Anda hanya mengandalkan `aws:PrincipalOrgID` untuk mengizinkan akses, federasi `AssumeRoleWithSAML` dan `AssumeRoleWithWebIdentity` panggilan ditolak secara implisit karena kunci kondisi tidak ada untuk non-prinsipal.AWS

Bagaimana AWS STS mengevaluasi kebijakan titik akhir VPC untuk pemanggil federasi

Saat pemanggil federasi memanggil `AssumeRoleWithSAML` atau `AssumeRoleWithWebIdentity` melalui titik akhir VPC, berikut ini berlaku:

- Penelepon bukanlah kepala sekolah AWS . Kunci kondisi seperti `aws:PrincipalOrgIDaws:PrincipalAccount`,, dan tidak `aws:PrincipalArn` tersedia dalam konteks permintaan.
- Penelepon federasi tidak memiliki kunci kondisi `aws:PrincipalIsAWSService` dalam konteks permintaan.
- Peran yang diasumsikan adalah AWS sumber daya. Resource-based kunci kondisi seperti `aws:ResourceOrgID` dan `aws:ResourceAccount` tersedia dan merujuk ke peran target.
- Kebijakan kepercayaan peran tetap menjadi gerbang otorisasi utama untuk akses federasi. Kebijakan titik akhir VPC menyediakan batas tingkat jaringan tambahan.

Sebaiknya gunakan `aws:ResourceOrgID` atau `aws:ResourceAccount` saat menulis pernyataan kebijakan titik akhir VPC yang harus berlaku untuk pemanggil federasi, karena pemanggil ini tidak memiliki kunci kondisi berbasis utama yang tersedia.

Kunci kondisi tersedia untuk AWS STS Kebijakan VPC endpoint

Tabel berikut menunjukkan kunci kondisi yang umum digunakan dan ketersediaannya dalam konteks permintaan untuk setiap jenis pemanggil saat membuat permintaan melalui titik akhir AWS STS VPC. Kunci kondisi tambahan tersedia di luar yang tercantum di sini.

Ketersediaan kunci kondisi berdasarkan jenis penelepon

Kunci syarat	Prinsipal yang diautentikasi AWS	Penelepon federasi	Deskripsi
`aws:PrincipalOrgID`	Ya	Tidak	ID organisasi dari kepala sekolah yang menelepon
`aws:PrincipalAccount`	Ya	Tidak	ID akun dari kepala sekolah yang menelepon
`aws:PrincipalArn`	Ya	Tidak	ARN kepala sekolah yang menelepon

Kunci syarat	Prinsipal yang diautentikasi AWS	Penelepon federasi	Deskripsi
`aws:PrincipalIsAWSService`	Ya (dievaluasi menjadi salah)	Tidak (kunci tidak ada)	Apakah penelepon adalah kepala AWS layanan
`aws:ResourceOrgID`	Ya	Ya	ID organisasi dari akun yang memiliki sumber daya yang diminta
`aws:ResourceAccount`	Ya	Ya	ID akun yang memiliki sumber daya yang diminta

> **Note**
>
> Untuk prinsipal yang AWS diautentikasi (pengguna dan peran IAM), `aws:PrincipalIsAWSService` hadir dalam konteks permintaan dan dievaluasi menjadi salah. Untuk pemanggil federasi, kunci ini tidak ada sepenuhnya dari konteks permintaan. Kondisi yang memeriksa `"Bool": {"aws:PrincipalIsAWSService": "false"}` tidak cocok dengan pemanggil federasi karena kunci tidak ada.

Contoh: Izinkan semua AWS STS Tindakan untuk organisasi Anda

Kebijakan titik akhir berikut membatasi titik akhir AWS STS VPC Anda ke organisasi Anda sambil mendukung akses federasi. Ini memungkinkan semua AWS STS tindakan untuk prinsipal yang diautentikasi di organisasi Anda dan secara terpisah mengizinkan pemanggil federasi untuk mengambil peran dalam organisasi Anda. Penelepon federasi (`AssumeRoleWithSAML`dan`AssumeRoleWithWebIdentity`) memerlukan pernyataan terpisah karena mereka tidak memiliki `aws:PrincipalOrgID` konteks permintaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrganizationPrincipals",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

Pernyataan pertama digunakan `"Principal": {"AWS": "*"}` dengan untuk mengiz `aws:PrincipalOrgID` inkan prinsipal yang diautenti AWS kasi dari organisasi Anda. Pernyataan kedua digunakan `"Principal": "*"` untuk mencocokkan pemanggil federasi dan membatasi peran target untuk digunakan organisasi Anda. `aws:ResourceOrgID` Kebijakan kepercayaan peran tetap menjadi kontrol utama di mana identitas federasi dapat mengambil peran mana.

Untuk informasi selengkapnya tentang menerapkan kontrol perimeter jaringan, lihat Mem bangun perimeter data AWS dan contoh kebijakan perimeter data di GitHub situs web.

Contoh: Izinkan semua AWS STS tindakan untuk akun tertentu

Kebijakan titik akhir berikut mengizinkan semua AWS STS tindakan untuk kepala sekolah di akun tertentu. Gunakan kebijakan ini jika akun Anda bukan bagian dari organisasi. Penelepon federasi

diizinkan dalam pernyataan terpisah karena mereka tidak memiliki `aws:PrincipalAccount` konteks permintaan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificAccountPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        }
    ]
}
```

Pernyataan pertama menggunakan `"Principal": {"AWS": "*"}` with `aws:PrincipalAccount` untuk mengizinkan prinsipal yang di AWS autentikasi dari akun yang ditentukan. Pernyataan kedua digunakan `"Principal": "*"` untuk mencocokkan pemanggil federasi dan membatasi peran target ke akun yang sama menggunakan. `aws:ResourceAccount` Kebijakan kepercayaan peran tetap menjadi kontrol utama di mana identitas federasi dapat mengambil peran mana.

Contoh: Batasi ke spesifik AWS STS tindakan

Kebijakan titik akhir berikut hanya mengizinkan `AssumeRole` untuk prinsipal yang diautentikasi dan `AssumeRoleWithSAML` untuk pemanggil federasi, keduanya dicakup untuk peran dalam organisasi Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRoleByOrgPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowSAMLFederation",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "sts:AssumeRoleWithSAML",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
```

```
    ]
}
```

Kebijakan ini memblokir AWS STS tindakan lain seperti`GetCallerIdentity`,`GetSessionToken`, dan `AssumeRoleWithWebIdentity` melalui titik akhir ini. Sesuaikan `Action` elemen agar sesuai dengan kebutuhan Anda.

Contoh: Menolak akses non-organisasi sambil mengizinkan federasi

Kebijakan titik akhir berikut menggunakan penolakan eksplisit untuk memblokir prinsipal yang diautentikasi di luar organisasi Anda, sambil mempertahankan akses untuk pemanggil federasi. Pendekatan ini dimulai dengan izin yang luas dan menambahkan pernyataan penolakan yang ditargetkan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAll",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "DenyNonOrgPrincipals",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

Pernyataan penolakan menggunakan`"Principal": {"AWS": "*"}`, yang hanya mencakup prinsipal yang diautentikasi AWS . Penelepon federasi (SAML dan OIDC) bukan prinsipal dan tidak AWS dicocokkan oleh `Principal` elemen ini, jadi penolakan tidak berlaku untuk mereka. Pendekatan ini menghindari perlunya kompleks `Null` atau `Bool` kondisi untuk mengukir pengecualian untuk penelepon federasi.

> **Note**
>
> Pernyataan pertama memungkinkan semua tindakan untuk semua kepala sekolah. Penolakan dalam pernyataan kedua diutamakan untuk prinsipal yang diautentikasi di AWS luar organisasi Anda. Penelepon federasi diizinkan oleh pernyataan pertama dan tidak terpengaruh oleh penolakan karena `Principal` elemen dalam pernyataan penolakan tidak cocok dengan mereka.

Gunakan tombol kondisi VPC untuk mengontrol akses federasi

Pengguna federasi dapat mengambil peran menggunakan `AssumeRoleWithSAML` atau `AssumeRoleWithWebIdentity` melalui titik akhir VPC. Saat melakukannya, Anda dapat menggunakan kunci VPC-specific kondisi dalam kebijakan kepercayaan peran atau dalam kebijakan kontrol sumber daya (RCP) untuk membatasi asal permintaan ini. Ini memberikan batas tingkat jaringan selain kontrol tingkat identitas dalam kebijakan kepercayaan.

Topik

- Cara kerja kunci kondisi VPC untuk permintaan federasi
- Contoh: Batasi federasi OIDC ke VPC tertentu
- Contoh: Batasi federasi SAMP ke titik akhir VPC tertentu
- Contoh: Batasi federasi OIDC dengan rentang IP pribadi dalam VPC
- Contoh: Batasi federasi SAMP ke beberapa VPC
- Contoh: Menggabungkan VPC dan kondisi khusus penyedia
- Contoh: Gunakan RCP untuk membatasi akses federasi ke VPC tertentu

Cara kerja kunci kondisi VPC untuk permintaan federasi

Kunci kondisi yang tersedia dalam konteks permintaan bergantung pada jalur jaringan permintaan federasi.

Ketersediaan kunci kondisi VPC berdasarkan jalur jaringan

Kunci syarat	Melalui titik akhir VPC	Melalui internet publik	Deskripsi
`aws:SourceVpc`	Ya	Tidak	ID VPC yang dilalui permintaan
`aws:SourceVpcArn`	Ya	Tidak	ARN dari VPC yang dilalui permintaan
`aws:SourceVpce`	Ya	Tidak	ID titik akhir VPC yang dilalui permintaan
`aws:VpcSourceIp`	Ya	Tidak	Alamat IP pribadi penelepon dalam VPC
`aws:SourceIp`	Tidak	Ya	Alamat IP publik penelepon

Jika kebijakan kepercayaan Anda menggunakan`aws:SourceVpc`,`aws:SourceVpcArn`, atau `aws:SourceVpce` dalam `Allow` pernyataan dan permintaan tidak datang melalui titik akhir VPC, kondisi tersebut tidak cocok dan secara AWS implisit menolak permintaan tersebut. Ini secara efektif membutuhkan permintaan gabungan untuk melintasi titik akhir VPC agar berhasil.

> **Important**
>
> Untuk permintaan yang dibuat melalui titik akhir VPC, `aws:SourceIp` tidak diisi. Gunakan `aws:VpcSourceIp` untuk IP-based pembatasan sebagai gantinya.

Kami merekomendasikan penggunaan `aws:SourceVpcArn` alih-alih `aws:SourceVpc` dalam kebijakan kepercayaan. ID VPC unik secara regional tetapi tidak unik secara global, sedangkan `aws:SourceVpcArn` mencakup Wilayah dan akun, memberikan identifikasi VPC yang unik secara global.

> **Note**
>
> Nilai audience (`aud`) dalam contoh OIDC berikut digunakan`sts.amazonaws.com`, yang merupakan default untuk Amazon EKS. Untuk penyedia OIDC lainnya, ganti ini dengan nilai audiens yang dikonfigurasi untuk aplikasi Anda (misalnya, ID klien atau URI aplikasi).

Contoh: Batasi federasi OIDC ke VPC tertentu

Kebijakan kepercayaan berikut memungkinkan penyedia OIDC untuk mengambil peran hanya ketika permintaan berasal dari VPC tertentu melalui titik akhir VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

AWS secara implisit menolak permintaan dari luar VPC yang ditentukan karena kondisinya tidak cocok. `aws:SourceVpcArn` Tidak diperlukan pernyataan penolakan eksplisit.

Contoh: Batasi federasi SAMP ke titik akhir VPC tertentu

Kebijakan kepercayaan berikut memungkinkan penyedia SAMP untuk mengambil peran hanya ketika permintaan datang melalui titik akhir VPC tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromVpce",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpce": "vpce-0abcdef1234567890"
                }
            }
        }
    ]
}
```

Menggunakan `aws:SourceVpce` alih-alih `aws:SourceVpc` memberikan kontrol yang lebih terperinci ketika Anda memiliki beberapa titik akhir VPC di VPC yang sama dan ingin membatasi akses ke titik akhir tertentu.

Contoh: Batasi federasi OIDC dengan rentang IP pribadi dalam VPC

Kebijakan kepercayaan berikut membatasi federasi untuk permintaan yang berasal dari subnet tertentu dalam VPC. Gunakan `aws:VpcSourceIp` untuk IP-based pembatasan permintaan yang dibuat melalui titik akhir VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromSubnet",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
```

```
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "IpAddress": {
                    "aws:VpcSourceIp": "10.0.1.0/24"
                }
            }
        }
    ]
}
```

> **Important**
>
> Jangan gunakan `aws:SourceIp` dengan rentang IP pribadi untuk permintaan melalui titik akhir VPC. `aws:SourceIp`Kuncinya tidak tersedia untuk permintaan titik akhir VPC. Gunakan `aws:VpcSourceIp` sebagai gantinya.

Contoh: Batasi federasi SAMP ke beberapa VPC

Kebijakan kepercayaan berikut memungkinkan penyedia SAMP untuk mengambil peran dari salah satu dari beberapa VPC. Ini berguna ketika organisasi Anda memiliki beberapa VPC di berbagai lingkungan atau Wilayah yang memerlukan akses gabungan ke peran yang sama.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromMultipleVpcs",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpcArn": [
                        "arn:aws:ec2:us-west-2:111122223333:vpc/vpc-111aaa11",
                        "arn:aws:ec2:us-east-1:111122223333:vpc/vpc-222bbb22"
                    ]
```

```
                }
            }
        }
    ]
}
```

Contoh: Menggabungkan VPC dan kondisi khusus penyedia

Anda dapat menggabungkan kunci kondisi VPC dengan kondisi khusus penyedia seperti audience (aud) dan subject (sub) untuk kontrol akses berbutir halus. Kebijakan kepercayaan berikut membatasi asumsi peran untuk pengguna federasi tertentu dan mengharuskan permintaan berasal dari VPC tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCUserFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "idp.example.com:sub": "user123",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

Ini menggabungkan kontrol tingkat identitas (membatasi untuk pengguna federasi tertentu) dengan kontrol tingkat jaringan (membatasi ke VPC tertentu), memberikan pertahanan secara mendalam.

Contoh: Gunakan RCP untuk membatasi akses federasi ke VPC tertentu

Kebijakan kontrol sumber daya (RCP) berikut menolak `AssumeRoleWithSAML` dan `AssumeRoleWithWebIdentity` meminta kecuali mereka berasal dari VPC tertentu. Tidak seperti

kebijakan kepercayaan yang berlaku per peran, kebijakan kontrol sumber daya (RCP) berlaku di semua peran di akun target. Ini memberikan batas jaringan terpusat.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyFederatedAccessOutsideVpc",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithWebIdentity",
                "sts:AssumeRoleWithSAML"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "Null": {
                    "aws:SourceVpcArn": "false"
                }
            }
        }
    ]
}
```

`Null`Kondisi memastikan penolakan hanya berlaku ketika `aws:SourceVpcArn` ada dalam konteks permintaan (yaitu, permintaan datang melalui titik akhir VPC tetapi dari VPC yang salah). Tanpa itu, permintaan melalui internet publik (di mana tidak `aws:SourceVpcArn` ada) juga akan ditolak oleh`StringNotEquals`. Jika Anda menghapus `Null` cek, penolakan juga berlaku untuk permintaan yang tidak melintasi titik akhir VPC apa pun, termasuk permintaan melalui internet publik. Kami merekomendasikan pengujian secara menyeluruh sebelum menghapus `Null` pemeriksaan di lingkungan produksi.

AWS layanan yang bekerja dengan IAM

AWS Layanan yang tercantum di bawah ini dikelompokkan menurut abjad dan mencakup informasi tentang fitur IAM apa yang mereka dukung:

- Layanan — Anda dapat memilih nama layanan untuk melihat AWS dokumentasi tentang otorisasi IAM dan akses untuk layanan tersebut.
- Tindakan – Anda dapat menentukan tindakan individu dalam kebijakan. Jika layanan tidak mendukung fitur ini, maka Semua tindakan dipilih di editor visual. Dalam dokumen kebijakan JSON, Anda harus menggunakan `*` di elemen `Action`. Untuk daftar tindakan di setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.
- Resource-level izin — Anda dapat menggunakan ARN untuk menentukan sumber daya individual dalam kebijakan. Jika layanan tidak mendukung fitur ini, maka Semua sumber daya dipilih di editor visual kebijakan. Dalam dokumen kebijakan JSON, Anda harus menggunakan `*` di elemen `Resource`. Beberapa tindakan, seperti tindakan `List*` , tidak mendukung penetapan ARN karena dirancang untuk mengembalikan beberapa sumber daya. Jika suatu layanan mendukung fitur ini untuk beberapa sumber daya tetapi tidak yang lain, itu ditunjukkan oleh Partial dalam tabel. Lihat dokumentasi untuk layanan tersebut untuk informasi selengkapnya.
- Resource-based kebijakan — Anda dapat melampirkan kebijakan berbasis sumber daya ke sumber daya dalam layanan. Resource-based kebijakan mencakup `Principal` elemen untuk menentukan identitas IAM mana yang dapat mengakses sumber daya tersebut. Untuk informasi selengkapnya, lihat Identity-based kebijakan dan kebijakan berbasis sumber daya.
- ABAC (otorisasi berdasarkan tag) - Untuk mengontrol akses berdasarkan tag, Anda memberikan informasi tag dalam elemen kondisi kebijakan menggunakan`aws:ResourceTag/`*`key-name`*`,aws:RequestTag/`*`key-name`*, atau kunci `aws:TagKeys` kondisi. Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

  Untuk informasi selengkapnya tentang mendefinisikan izin berdasarkan atribut seperti tanda, lihat Tentukan izin berdasarkan atribut dengan otorisasi ABAC. Untuk melihat tutorial dengan langkah-langkah untuk menyiapkan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC).
- Kredensi sementara — Anda dapat menggunakan kredensil jangka pendek yang Anda peroleh saat masuk menggunakan Pusat Identitas IAM, pengelola akses akun, beralih peran di konsol, atau yang Anda hasilkan menggunakan AWS STS di atau API. AWS CLI AWS Anda dapat mengakses layanan dengan nilai Tidak hanya saat menggunakan kredensi pengguna IAM jangka panjang Anda. Ini termasuk nama pengguna dan kata sandi atau access key pengguna. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.
- Service-linked peran — Peran terkait layanan adalah jenis peran layanan khusus yang memberikan izin layanan untuk mengakses sumber daya di layanan lain atas nama Anda. Pilih tautan Ya atau Sebagian untuk melihat dokumentasi layanan yang mendukung peran ini. Kolom

ini tidak menunjukkan apakah layanan menggunakan peran layanan standar. Untuk informasi selengkapnya, lihat Service-linked peran.

- Informasi selengkapnya – Jika layanan tidak sepenuhnya mendukung fitur, Anda dapat meninjau catatan kaki untuk entri untuk melihat pembatasan dan tautan ke informasi terkait.

Layanan yang bekerja dengan IAM

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Manajer akses akun	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	Ya
AWS Account Management	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Rekomendasi Tindakan	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Activate Console	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Operasi Amazon AI	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Amplify Admin	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Amplify	✓Ya	✓Ya	✗Tidak	⚠Sebagian	✓Ya	✗Tidak
AWS Amplify Pembuat UI	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Apache Kafka API untuk kluster MSK Amazon	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon API Gateway	✓Ya	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya
Manajemen Amazon API Gateway	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Manajemen Amazon API Gateway V2	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Studio Aplikasi	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS App2Container	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS AppConfig	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS AppFabric	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon AppFlow	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon AppIntegrations	Ya	Ya	Tidak	Ya	Ya	Ya
Penskalaan Otomatis Aplikasi	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Aplikasi Discovery Arsenal	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Application Discovery Service	Ya	Tidak	Tidak	Tidak	Ya	Ya
AWS Transform MGN	Ya	Ya	Tidak	Ya	Ya	Ya
Pengontrol Pemulihan Aplikasi Amazon - Zonal Shift	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Layanan Transformasi Aplikasi	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS App Mesh	Ya	Ya	Tidak	Ya	Ya	Ya
AWS App Mesh Pratinjau	Ya	Ya	Tidak	Tidak	Ya	Ya
AWS Pelari Aplikasi	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon AppStream 2.0	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS AppSync	Ya	Ya	Tidak	Ya	Ya	Tidak
Sakelar Wilayah Amazon ARC	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Artifact	Ya	Ya	Tidak	Tidak	Ya	Tidak
Amazon Athena	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Audit Manager	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Aurora DSQL	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Auto Scaling	Ya	Tidak	Tidak	Tidak	Ya	Ya
AWS Pertukaran Data B2B	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Backup	Ya	Ya	Ya	Ya	Ya	Ya
AWS Backup Gerbang	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Backup Cari	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Backup penyimpanan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Batch	Ya	Sebagian	Tidak	Ya	Ya	Ya
Amazon Bedrock	Ya	Ya	Tidak	Ya	Ya	Tidak
Agen Batuan Dasar Amazon	Ya	Ya	Tidak	Sebagian	Ya	Tidak
AWS Manajemen Penagihan dan Biaya	Ya	Ya	Tidak	Ya	Ya	Ya
Dasbor Manajemen Penagihan dan Biaya AWS	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Manajemen Penagihan dan Biaya Ekspor Data	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Manajemen Penagihan dan Biaya Kalkulator Harga	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Manajemen Penagihan dan Biaya Tindakan yang Direkomendasikan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Billing Conductor	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Braket	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Layanan Anggaran	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Certificate Manager (ACM)	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Q Developer dalam aplikasi obrolan	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Chime	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Clean Rooms	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Clean Rooms ML	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Client VPN	Ya	Ya	Tidak	Tidak	Ya	Ya
AWS Cloud9	Ya	Ya	Ya	Ya	Ya	Ya
AWS Cloud Kontrol API	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Amazon Cloud Directory	Ya	Ya	Tidak	Tidak	Ya	Tidak
CloudFormation	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudFront	Ya	Ya	Tidak	Sebagian	Ya	Ya
Amazon CloudFront KeyValueStore	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CloudHSM	Ya	Ya	Tidak	Ya	Ya	Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Cloud Map	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudSearch	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS CloudShell	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS CloudTrail	Ya	Ya	Sebagian (Info)	Ya	Ya	Ya
AWS CloudTrail Data	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudWatch	Ya	Ya	Tidak	Ya	Ya	Sebagian (Info)
Wawasan CloudWatch Aplikasi Amazon	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Sinyal CloudWatch Aplikasi Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Monitor CloudWatch Internet Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
CloudWatch Log Amazon	Ya	Ya	Ya	Sebagian	Ya	Ya
Monitor Sintetis CloudWatch Jaringan Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Manajer Akses CloudWatch Observabilitas Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Layanan Admin CloudWatch Observabilitas Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon CloudWatch RUM	Ya	Ya	Tidak	Ya	Ya	Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon CloudWatch Synthetics	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodeArtifact	Ya	Ya	Ya	Ya	Ya	Tidak
AWS CodeBuild	Ya	Ya	Ya (Info)	Sebagian (Info)	Ya	Tidak
Amazon CodeCatalyst	Ya	Ya	Tidak	Ya	Ya	Ya
AWS CodeCommit	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodeConnections	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodeDeploy	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodeDeploy layanan perintah host aman	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Amazon CodeGuru Profiler	Ya	Ya	Tidak	Ya	Ya	Ya
CodeGuru Peninjau Amazon	Ya	Ya	Tidak	Ya	Ya	Ya
CodeGuru Keamanan Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS CodePipeline	Ya	Sebagian	Tidak	Ya	Ya	Tidak
AWS CodeStar	Ya	Sebagian	Tidak	Ya	Ya	Tidak
AWS CodeStar Koneksi	Ya	Ya	Tidak	Ya	Ya	Ya
AWS CodeStar Pemberitahuan	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon CodeWhisperer	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Cognito	Ya	Ya	Tidak	Ya	Ya	Ya
Sinkronisasi Amazon Cognito	Ya	Ya	Tidak	Tidak	Ya	Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Kumpulan pengguna Amazon Cognito	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Comprehend	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Comprehend Medical	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Compute Optimizer	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✓Ya
AWS Config	✓Ya	⚠Sebagian (Info)	✗Tidak	✓Ya	✓Ya	✓Ya
Pelanggan Amazon Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Kasus Pelanggan Amazon Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Profil Pelanggan Pelanggan Amazon Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Kampanye Keluar Pelanggan Amazon Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
ID Suara Pelanggan Amazon Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Connect Health	✓Ya	✓Ya	✗Tidak	⚠Sebagian	✓Ya	✗Tidak
AWS Console Mobile Application	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Penagihan Konsolidasi	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Katalog Kontrol	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Control Tower	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Cost and Usage Report	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Cost Explorer	Ya	Ya	Tidak	Ya	Ya	Tidak
Hub Optimisasi Biaya AWS	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Layanan Verifikasi Pelanggan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Database Migration Service	Ya	Ya	Tidak (Info)	Ya	Ya	Ya
Layanan Metadata Kueri Basis Data	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Data Exchange	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Data Lifecycle Manager	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Data Pipeline	Ya	Ya	Tidak	Sebagian	Ya	Tidak
AWS DataSync	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon DataZone	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Batas Waktu Cloud	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS DeepRacer	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Detective	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Device Farm	Ya	Ya	Tidak	Ya	Ya	Ya
DevOpsGuru Amazon	Ya	Ya	Tidak	Tidak	Ya	Ya
AWS Alat diagnostik	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Direct Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Directory Service	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Directory Service Data	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Cluster Elastis Amazon DocumentDB	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Akselerator Amazon DynamoDB (DAX)	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✓Ya
Amazon DynamoDB	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya
Amazon Elastic Compute Cloud (Amazon EC2)	✓Ya	⚠Sebagian	✗Tidak	✓Ya	✓Ya	⚠Sebagian (Info)
Amazon EC2 Auto Scaling	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
EC2 Image Builder	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Connect Instans Amazon EC2	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Elastic Beanstalk	✓Ya	⚠Sebagian	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Elastic Block Store (Amazon EBS)	✓Ya	⚠Sebagian	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Elastic Container Registry (Amazon ECR)	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya
Registri Kontainer Elastis Amazon Publik (Amazon ECR Publik)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon Elastic Container Service (Amazon ECS)	✓Ya	⚠Sebagian (Info)	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Elastic Disaster Recovery	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Elastic File System (Amazon EFS)	✓Ya	✓Ya	✓Ya	⚠Sebagian	✓Ya	✓Ya
Amazon Elastic Kubernetes Service (Amazon EKS)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Auth Layanan Amazon Elastic Kubernetes (Amazon EKS)	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Elastic Load Balancing	✓Ya	⚠Sebagian	✗Tidak	⚠Sebagian	✓Ya	✓Ya
Layanan Amazon Elastic VMware	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon ElastiCache	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Peralatan Elemental dan Layanan Aktivasi Perangkat Lunak	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Peralatan dan Perangkat Lunak Elemental	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Elemental MediaConnect	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✓Ya
AWS Elemental MediaConvert	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Elemental MediaLive	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Elemental MediaPackage	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	⚠Sebagian (Info)

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Elemental MediaPackage V2	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Elemental MediaPackage VOD	Ya	Ya	Tidak	Ya	Ya	Sebagian (Info)
AWS Elemental MediaTailor	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Kasus Dukungan Elemental	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Konten Dukungan Elemental	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Amazon EMR	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon EMR di EKS	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon EMR Tanpa Server	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Olah Pesan Pengguna Akhir SMS dan Voice V2	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Pesan Pengguna Akhir Sosial	Ya	Ya	Tidak	Ya	Ya	Ya
Resolusi Entitas AWS	Ya	Ya	Ya	Ya	Ya	Tidak
Amazon EventBridge	Ya	Ya	Ya	Ya	Ya	Tidak
EventBridge Pipa Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
EventBridge Penjadwal Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
EventBridge Skema Amazon	Ya	Ya	Ya	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Layanan Injeksi Kesalahan	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon FinSpace	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon FinSpace API	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Firewall Manager	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	⚠Sebagian
Fleet Hub for AWS IoT Device Management	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Forecast	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Fraud Detector	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
FreeRTOS	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Tingkat Gratis	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon FSx	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
GameLift Server Amazon	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
GameLift Aliran Amazon	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Global Accelerator	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Glue	✓Ya	✓Ya	⚠Sebagian	⚠Sebagian	✓Ya	✗Tidak
AWS Glue DataBrew	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Ground Station	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Pelabelan Ground Truth Amazon	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon GuardDuty	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Health API Dan Pemberitahuan	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS HealthImaging	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS HealthLake	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS HealthOmics	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IAM Identity Center	Ya	Ya	Tidak	Sebagian	Ya	Ya
Direktori Pusat Identitas IAM	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Toko Identitas Pusat Identitas IAM	Ya	Ya	Tidak	Tidak	Ya	Tidak
Layanan IAM Identity Center OIDC	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Identity and Access Management (IAM)	Ya	Ya	Sebagian (Info)	Sebagian (Info)	Sebagian (Info)	Tidak
AWS Identity and Access Management dan Access Analyzer	Ya	Ya	Tidak	Ya	Ya	Sebagian
AWS Identity and Access Management Peran Di Mana Saja	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Auth Toko Identitas	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Sinkronisasi Identitas	Ya	Ya	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Import/Export	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Amazon Inspector	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Inspector Klasik	Ya	Tidak	Tidak	Tidak	Ya	Ya
Amazon InspectorScan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Layanan Video Interaktif Amazon	Ya	Ya	Tidak	Ya	Ya	Ya
Obrolan Layanan Video Interaktif Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Invoicing	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT Analytics	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT	Ya	Ya	Sebagian (Info)	Ya	Ya	Tidak
AWS IoT Core Penasihat Perangkat	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT Penguji Perangkat	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS IoT Events	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT FleetWise	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT Greengrass	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS IoT Greengrass V2	Ya	Ya	Tidak	Sebagian	Ya	Tidak
AWS IoT Lowongan DataPlane	Ya	Ya	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS IoT Layanan Integrasi Terkelola	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS IoT SiteWise	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS IoT TwinMaker	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS IoT Wireless	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Kendra	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Peringkat Cerdas Amazon Kendra	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Key Management Service (AWS KMS)	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya
Amazon Keyspaces (untuk Apache Cassandra)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Layanan Dikelola Amazon untuk Apache Flink	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Layanan Dikelola Amazon untuk Apache Flink V2	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Data Firehose	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Kinesis Data Streams	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✗Tidak
Aliran Video Amazon Kinesis	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Lake Formation	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✓Ya
AWS Lambda	✓Ya	✓Ya	✓Ya	⚠Sebagian (Info)	✓Ya	⚠Sebagian (Info)

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Launch Wizard	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Lex	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Lex V2	✓Ya	✓Ya	✓Ya	✓Ya	✗Tidak	✓Ya
AWS License Manager	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS License Manager Manajer Langganan Linux	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS License Manager Langganan Pengguna	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✓Ya
Amazon Lightsail	✓Ya	⚠Sebagian (Info)	✗Tidak	⚠Sebagian (Info)	✓Ya	✓Ya
Amazon Location Service	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Peta Amazon Location Service	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Tempat Layanan Lokasi Amazon	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Rute Layanan Lokasi Amazon	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Lookout for Equipment	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Lookout for Metrics	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Machine Learning	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Macie	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Mainframe Modernization	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Mainframe Modernization Pengujian Aplikasi	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Blockchain yang Dikelola Amazon	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Kueri Blockchain yang Dikelola Amazon	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Managed Grafana	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Layanan Terkelola Amazon untuk Prometheus	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Managed Streaming for Apache Kafka (MSK)	✓Ya	✓Ya	⚠Sebagian (Info)	✓Ya	✓Ya	✓Ya
Amazon Managed Streaming for Kafka Connect	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Alur Kerja Terkelola Amazon untuk Apache Airflow	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Marketplace	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✓Ya
AWS Marketplace Katalog	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Marketplace Commerce Analytics	✓Ya	✗Tidak	✗Tidak	✗Tidak	✗Tidak	✗Tidak
AWS Marketplace Layanan Deployment	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Marketplace Penemuan	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Marketplace Entitlement Service	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Layanan Image Building	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Portal Manajemen AWS Marketplace	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Metering Service	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Marketplace Pribadi	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Integrasi Sistem Pengadaan	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Pelaporan	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Pelaporan Penjual	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Marketplace Wawasan Vendor	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon MediaImport	✓Ya	✗Tidak	✗Tidak	✗Tidak	✗Tidak	✗Tidak
Amazon MemoryDB	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Layanan Pengiriman Pesan Amazon	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Layanan Gateway Pesan Amazon	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Microservice Extractor for .NET	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Kredit Program Percepatan Migrasi	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS Migration Hub	Ya	Ya	Tidak	Tidak	Ya	Ya
AWS Migration Hub Orkestrator	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Migration Hub Refactor Spaces	Ya	Ya	Ya	Ya	Ya	Ya
AWS Migration Hub Rekomendasi Strategi	Ya	Tidak	Tidak	Tidak	Ya	Ya
Amazon Monitron	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon MQ	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Neptune	Ya	Ya	Tidak	Tidak	Ya	Ya
Multi-partypersetujuan	Ya	Ya	Tidak	Sebagian (Info)	Ya	Tidak
Analisis Amazon Neptunus	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Network Firewall	Ya	Ya	Tidak	Ya	Ya	Ya
Monitor Aliran Jaringan	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Network Manager	Ya	Ya	Tidak	Ya	Ya	Ya (Info)
AWS Network Manager Obrolan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Studio Amazon Nimble	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon Satu Perusahaan	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon OpenSearch	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
OpenSearchTertelan Amazon	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Tanpa OpenSearch Server	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
OpenSearch Layanan Amazon	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya
Database Oracle@AWS	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS OpsWorks	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS OpsWorks Manajemen Konfigurasi	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Organizations	✓Ya	✓Ya	✓Ya	✓Ya	✗Tidak	✓Ya
AWS Outposts	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Layanan Komputasi Paralel	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Manajemen akun Partner Central	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Penjualan Sentral Mitra	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Payment Cryptography	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Pembayaran	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Wawasan Performa	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Personalisasi	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon Pinpoint	Ya	Ya	Tidak	Ya	Ya	Tidak
Layanan Email Amazon Pinpoint	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Pinpoint SMS dan Layanan Suara	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Amazon Polly	Ya	Ya	Tidak	Tidak	Ya	Tidak
Daftar Harga AWS	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS 5G pribadi	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Private CA Konektor untuk Active Directory	Ya	Ya	Tidak	Ya	Ya	Tidak
Konektor untuk SCEP AWS Private CA	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Private Certificate Authority (AWS Private CA)	Ya	Ya	Ya	Ya	Ya	Tidak
AWS PrivateLink	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Proton	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Konsol Pesanan Pembelian	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon Q Bisnis	Ya	Ya	Tidak	Ya	Ya	Ya
Aplikasi Q Bisnis Amazon Q	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Q Developer	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Q di Connect	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon Quantum Ledger Database (Amazon QLDB)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Cepat	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
API Data Amazon RDS	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Otentikasi Amazon RDS IAM	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Tempat Sampah Daur Ulang	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Redshift	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
API Data Pergeseran Merah Amazon	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Redshift Tanpa Server	✓Ya	✓Ya	✓Ya (Info)	✓Ya	✓Ya	✗Tidak
Amazon Rekognition	✓Ya	✓Ya	⚠Sebagian (Info)	✓Ya	✓Ya	✗Tidak
Layanan Database Relasional Amazon (Amazon RDS) (Info)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS re:Post Pribadi	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Resilience Hub	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Resource Access Manager (AWS RAM)	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Penjelajah Sumber Daya AWS	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Resource Groups	Ya	Ya	Tidak	Ya	Sebagian (Info)	Ya
AWS Resource Groups Tagging API	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Portal Basis Pengetahuan Amazon RHEL	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS RoboMaker	Ya	Ya	Tidak	Ya	Ya	Ya
Rute Amazon 53	Ya	Ya	Tidak	Tidak	Ya	Tidak
Amazon Route 53 Domain	Ya	Tidak	Tidak	Tidak	Tidak	Tidak
Profil Amazon Route 53	Ya	Ya	Tidak	Ya	Ya	Tidak
Cluster Pemulihan Amazon Route 53	Ya	Ya	Tidak	Tidak	Ya	Tidak
Config Kontrol Pemulihan Amazon Route 53	Ya	Ya	Tidak	Ya	Ya	Tidak
Kesiapan Pemulihan Amazon Route 53	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon Route 53 Resolver	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Kain RTB	Ya	Ya	Tidak	Ya	Ya	Ya
Amazon S3 Ekspres	Ya	Ya	Ya	Ya	Ya	Tidak
Amazon Glacier	Ya	Ya	Ya	Tidak	Ya	Sebagian
Tabel Amazon S3	Ya	Ya	Ya	Tidak	Ya	Tidak
Vektor Amazon S3	Ya	Ya	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Amazon SageMaker AI	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	⚠Sebagian (Info)
Asisten ilmu data Amazon SageMaker AI	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Kemampuan geospasial Amazon SageMaker AI	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon SageMaker AI dengan MLFlow	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Savings Plans	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Secrets Manager	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✗Tidak
AWS Security Hub CSPM	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Respon Insiden Keamanan	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
Amazon Security Lake	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✓Ya
AWS Security Token Service (AWS STS)	✓Ya	⚠Sebagian (Info)	✗Tidak	✓Ya	⚠Sebagian (Info)	✗Tidak
AWS Serverless Application Repository	✓Ya	✓Ya	✓Ya	✗Tidak	✓Ya	✗Tidak
AWS Service Catalog	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Layanan untuk mengelola kemampuan pengalaman pengguna AWS Konsol	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Service Quotas	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Shield	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Shield direktur keamanan jaringan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Signer	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Sign-In	Ya	Tidak	Ya	Tidak	Ya	Tidak
Amazon SimpleDB	Ya	Ya	Tidak	Tidak	Ya	Tidak
Layanan Email Sederhana Amazon - Manajer Surat	Ya	Ya	Tidak	Ya	Ya	Ya
Layanan Email Sederhana Amazon (Amazon SES) v2	Ya	Sebagian (Info)	Ya	Ya	Sebagian (Info)	Ya
Amazon Simple Notification Service (Amazon SNS)	Ya	Ya	Ya	Ya	Ya	Tidak
Amazon Simple Queue Service (Amazon SQS)	Ya	Ya	Ya	Sebagian	Ya	Tidak
Amazon Simple Storage Service (Amazon S3)	Ya	Ya	Ya	Sebagian (Info)	Ya	Sebagian (Info)
Layanan Penyimpanan Sederhana Amazon (Amazon S3) Objek Lambda	Ya	Ya	Tidak	Tidak	Ya	Tidak
Amazon Simple Storage Service (Amazon S3) aktif AWS Outposts	Ya	Ya	Ya	Tidak	Ya	Ya
Amazon Simple Workflow Service (Amazon SWF)	Ya	Ya	Tidak	Ya	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS SimSpacePenenun	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Site-to-Site VPN	Ya	Ya	Tidak	Tidak	Ya	Ya
AWS Snowball Edge	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Snowball Edge Tepi	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Snowball Edge Device Management	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS SQL Workbench	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Step Functions	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Storage Gateway	Ya	Ya	Tidak	Ya	Ya	Tidak
Rantai Pasokan AWS	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Dukungan	Ya	Tidak	Tidak	Tidak	Ya	Ya
AWS Support App in Slack	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Dukungan Konsol	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Dukungan Rencana	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Keberlanjutan	Ya	Tidak	Tidak	Tidak	Ya	Tidak
AWS Systems Manager	Ya	Ya	Sebagian	Ya	Ya	Ya
AWS Systems Manager untuk SAP	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Systems Manager GUI Connect	Ya	Tidak	Tidak	Tidak	Ya	Tidak

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Manajer Insiden AWS Systems Manager	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya	✓Ya
Manajer Insiden AWS Systems Manager Kontak	✓Ya	✓Ya	✓Ya	✗Tidak	✓Ya	✗Tidak
AWS Systems Manager Pengaturan Cepat	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Penyunting Tag	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Pengaturan Pajak	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
AWS Pembangun Jaringan Telekomunikasi	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Amazon Texttract	✓Ya	✗Tidak	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Timestream	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
Influxdb Amazon Timestream	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya
AWS Tiros API (untuk Reachability Analyzer)	✓Ya	✗Tidak	✗Tidak	✗Tidak	✗Tidak	✗Tidak
Amazon Transcribe	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Transfer Family	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Transformasi	✓Ya	✓Ya	✗Tidak	✗Tidak	✓Ya	✗Tidak
Amazon Translate	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✗Tidak
AWS Trusted Advisor	⚠Sebagian (Info)	✓Ya	✗Tidak	✗Tidak	⚠Sebagian	✓Ya
AWS Pemberitahuan Pengguna	✓Ya	✓Ya	✗Tidak	✓Ya	✓Ya	✓Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
AWS Kontak Pemberitahuan Pengguna	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Langganan Pengguna	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Akses Terverifikasi AWS	Ya	Tidak	Tidak	Tidak	Ya	Tidak
Izin Terverifikasi Amazon	Ya	Ya	Tidak	Tidak	Ya	Tidak
Amazon Virtual Private Cloud (Amazon VPC)	Ya	Sebagian (Info)	Sebagian (Info)	Ya	Ya	Sebagian (Info)
Kisi VPC Amazon	Ya	Ya	Tidak	Ya	Ya	Ya
Layanan Kisi VPC Amazon	Ya	Ya	Tidak	Tidak	Ya	Tidak
AWS WAF	Ya	Ya	Tidak	Ya	Ya	Ya
AWS WAF Klasik	Ya	Ya	Tidak	Ya	Ya	Ya
AWS WAF Regional	Ya	Ya	Tidak	Ya	Ya	Ya
AWS Well-Architected Tool	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS Wickr	Ya	Ya	Tidak	Ya	Ya	Tidak
Amazon WorkMail	Ya	Ya	Tidak	Ya	Ya	Ya
Alur WorkMail Pesan Amazon	Ya	Ya	Tidak	Tidak	Ya	Tidak
Amazon WorkSpaces	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS WorkSpaces Instans Terkelola	Ya	Ya	Tidak	Ya	Ya	Tidak
Browser WorkSpaces Aman Amazon	Ya	Ya	Tidak	Ya	Ya	Ya

Layanan	Tindakan	Resource-level izin	Resource-based kebijakan	ABAC	Kredensial sementara	Service-linked peran
Klien WorkSpaces Tipis Amazon	Ya	Ya	Tidak	Ya	Ya	Tidak
AWS X-Ray	Ya	Sebagian (Info)	Tidak	Sebagian (Info)	Ya	Tidak

Informasi selengkapnya

AWS CloudTrail

CloudTrail mendukung kebijakan berbasis sumber daya pada penyimpanan data acara CloudTrail Lake, dasbor, dan saluran yang digunakan untuk integrasi dengan sumber peristiwa di luar. AWS

Amazon CloudWatch

CloudWatch peran terkait layanan tidak dapat dibuat menggunakan Konsol Manajemen AWS, dan hanya mendukung fitur Tindakan Alarm.

AWS CodeBuild

CodeBuild mendukung berbagi sumber daya lintas akun menggunakan AWS RAM.

CodeBuild mendukung ABAC untuk tindakan berbasis proyek.

AWS Config

AWS Config mendukung izin tingkat sumber daya untuk agregasi dan Aturan data multi-wilayah multi-akun. AWS Config Untuk daftar sumber daya yang didukung, lihat bagian Agregasi Multi-Account Multi-Region Data dan bagian AWS Config Aturan pada Panduan AWS Config API.

AWS Database Migration Service

Anda dapat membuat dan mengubah kebijakan yang dilampirkan ke kunci AWS KMS enkripsi yang Anda buat untuk mengenkripsi data yang dimigrasi ke titik akhir target yang didukung. Titik akhir target yang didukung termasuk Amazon Redshift dan Amazon S3. Untuk informasi selengkapnya, lihat Membuat dan Menggunakan AWS KMS Kunci untuk Mengenkripsi Data Target Amazon

Redshift dan AWS KMS Membuat Kunci untuk Mengenkripsi Objek Target Amazon S3 di Panduan Pengguna.AWS Database Migration Service

Amazon Elastic Compute Cloud

Peran terkait layanan Amazon EC2 hanya dapat digunakan untuk fitur berikut: Permintaan Instans Spot, Permintaan Armada Spot, Armada Amazon EC2, dan Peluncuran cepat untuk instansWindows.

Amazon Elastic Container Service

Hanya beberapa tindakan Amazon ECS yang mendukung izin tingkat sumber daya.

AWS Elemental MediaPackage

MediaPackage mendukung peran terkait layanan untuk memublikasikan log akses pelanggan ke CloudWatch tetapi tidak untuk tindakan API lainnya.

AWS Identity and Access Management

IAM hanya mendukung satu jenis kebijakan berbasis sumber daya yang disebut kebijakan kepercayaan peran, yang melekat pada peran IAM. Untuk informasi selengkapnya, lihat Berikan izin pengguna untuk beralih peran.

IAM mendukung kontrol akses berbasis tag untuk sebagian besar sumber daya IAM. Untuk informasi selengkapnya, lihat Tag untuk AWS Identity and Access Management sumber daya.

Hanya beberapa tindakan API untuk IAM yang dapat dipanggil dengan kredensil sementara. Untuk informasi selengkapnya, lihat Membandingkan opsi API Anda.

AWS IoT

Perangkat yang terhubung ke AWS IoT diautentikasi dengan menggunakan X.509 sertifikat atau menggunakan Identitas Amazon Cognito. Anda dapat melampirkan AWS IoT kebijakan ke X.509 sertifikat atau Identitas Amazon Cognito untuk mengontrol apa yang diizinkan oleh perangkat. Untuk informasi selengkapnya, lihat Keamanan dan Identitas untuk AWS IoT dalam Panduan Developer AWS IoT .

AWS Lambda

Lambda mendukung kontrol akses berbasis atribut (ABAC) untuk fungsi, pemetaan sumber peristiwa, dan konfigurasi penandatanganan kode. Lapisan tidak didukung. Untuk informasi selengkapnya, lihat Menggunakan kontrol akses berbasis atribut di Lambda.

Lambda tidak memiliki peran terkait layanan, tetapi Lambda @Edge memilikinya. Untuk informasi selengkapnya, lihat Service-Linked Peran untuk Lambda @Edge di Panduan CloudFront Pengembang Amazon.

Amazon Lightsail

Lightsail sebagian mendukung izin tingkat sumber daya dan ABAC. Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon Lightsail.

Arus yang Dikelola Amazon untuk Apache Kafka (MSK)

Anda dapat melampirkan kebijakan klaster ke kluster MSK Amazon yang telah dikonfigurasi untuk konektivitas multi-VPC.

Multi-party persetujuan

Untuk sumber identitas Multi-party persetujuan dan tim persetujuan, Anda dapat menggunakan ketiga kunci kondisi ABAC:`aws:ResourceTag/`*`key-name`*`,aws:RequestTag/`*`key-name`*, dan`aws:TagKeys`.

Untuk sesi, Anda hanya dapat menggunakan tombol `aws:ResourceTag/`*`key-name`* kondisi karena:

- Sesi bukan sumber daya yang dapat diberi tag secara langsung.
- Sesi mewarisi tag dari tim persetujuan yang terkait dengannya.

AWS Network Manager

AWS Cloud WAN juga mendukung peran terkait layanan. Untuk informasi selengkapnya, lihat peran terkait layanan AWS Cloud WAN di Panduan Amazon VPC AWS Cloud WAN.

Amazon Redshift Tanpa Server

Redshift Serverless mendukung kebijakan berbasis sumber daya untuk berbagi snapshot.

Amazon Relational Database Service

Amazon Aurora adalah mesin database relasional yang dikelola sepenuhnya yang kompatibel dengan MySQL dan PostgreSQL. Anda dapat memilih Aurora MySQL atau Aurora PostgreSQL sebagai opsi mesin DB saat menyiapkan server database baru melalui Amazon RDS. Untuk

informasi selengkapnya, lihat Manajemen identitas dan akses untuk Amazon Aurora di Panduan Pengguna Amazon Aurora.

Amazon Rekognition

Resource-based kebijakan hanya didukung untuk menyalin model Label Kustom Rekognition Amazon.

AWS Resource Groups

Pengguna dapat mengambil peran dengan kebijakan yang memungkinkan operasi Resource Groups.

Amazon SageMaker AI

Service-linked peran saat ini tersedia untuk pekerjaan SageMaker AI Studio dan pelatihan SageMaker AI.

AWS Security Token Service

AWS STS tidak memiliki “sumber daya,” tetapi memungkinkan pembatasan akses dengan cara yang mirip dengan pengguna. Untuk informasi selengkapnya, lihat Menolak akses ke kredensial keamanan sementara menurut nama.

Hanya beberapa operasi API untuk panggilan AWS STS dukungan dengan kredensil sementara. Untuk informasi selengkapnya, lihat Membandingkan opsi API Anda.

Layanan Email Sederhana Amazon

Anda hanya dapat menggunakan izin tingkat sumber daya dalam pernyataan kebijakan yang merujuk pada tindakan yang terkait dengan pengiriman email, seperti atau. `ses:SendEmail` `ses:SendRawEmail` Untuk pernyataan kebijakan yang mengacu pada tindakan lainnya, elemen Sumber daya hanya dapat berisi `*`.

Hanya Amazon SES API yang mendukung kredensi keamanan sementara. Antarmuka Amazon SES SMTP tidak mendukung kredensial SMTP yang berasal dari kredensial keamanan sementara.

Amazon Simple Storage Service

Amazon S3 mendukung otorisasi berbasis tag untuk bucket direktori, sumber daya objek, Lensa Penyimpanan Amazon S3, grup Lensa Penyimpanan, dan titik akses.

Amazon S3 mendukung peran terkait layanan untuk Lensa Penyimpanan Amazon S3.

AWS Trusted Advisor

Akses API ke Trusted Advisor adalah melalui Dukungan API dan dikendalikan oleh kebijakan Dukungan IAM.

Cloud Privat Virtual Amazon

Amazon VPC mendukung melampirkan kebijakan sumber daya tunggal ke titik akhir VPC untuk membatasi apa yang dapat diakses melalui titik akhir tersebut. Untuk informasi selengkapnya tentang penggunaan kebijakan berbasis sumber daya untuk mengontrol akses ke sumber daya dari titik akhir VPC Amazon tertentu, lihat Mengontrol akses ke layanan menggunakan kebijakan titik akhir di Panduan.AWS PrivateLink

AWS X-Ray

X-Ray tidak mendukung izin tingkat sumber daya untuk semua tindakan.

X-Ray mendukung kontrol akses berbasis tag untuk grup dan aturan pengambilan sampel.

AWS Tanda Tangan Versi 4 untuk permintaan API

> **Important**
>
> Jika Anda menggunakan alat AWS SDK (lihat Kode Sam pel dan Perpustakaan) atau AWS Command Line Interface (AWS CLI) untuk mengirim permintaan API AWS, Anda dapat melewati proses tanda tangan, karena klien SDK dan CLI mengotentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Kecuali Anda memiliki alasan bagus untuk tidak melakukannya, kami sarankan Anda selalu menggunakan SDK atau CLI.
> Di Wilayah yang mendukung beberapa versi tanda tangan, permintaan penandatanganan secara manual berarti Anda harus menentukan versi tanda tangan mana yang akan digunakan. Saat Anda memberikan permintaan ke Multi-Region Access Points, SDK dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

Informasi otentikasi yang Anda kirim dalam permintaan harus menyertakan tanda tangan. AWS Signature Version 4 (SIGv4) adalah protokol pen AWS andatanganan untuk menambahkan informasi otentikasi ke permintaan AWS API.

Anda tidak menggunakan kunci akses rahasia untuk menandatangani permintaan API. Sebagai gantinya, Anda menggunakan proses penandatanganan SIGv4. Permintaan penandatanganan melibatkan:

1. Membuat permintaan kanonik berdasarkan detail permintaan.
2. Menghitung tanda tangan menggunakan AWS kredensibilitas Anda.
3. Menambahkan tanda tangan ini ke permintaan sebagai header Otorisasi.

AWS kemudian mereplikasi proses ini dan memverifikasi tanda tangan, memberikan atau menolak akses yang sesuai.

SigV4 simetris mengharuskan Anda untuk mendapatkan kunci yang dicakup ke satu AWS layanan, dalam satu AWS wilayah, pada hari tertentu. Ini membuat kunci dan tanda tangan yang dihitung berbeda untuk setiap wilayah, yang berarti Anda harus mengetahui wilayah tujuan tanda tangan tersebut.

Asymmetric Signature Version 4 (SigV4a) adalah ekstensi yang mendukung penandatanganan dengan algoritma baru, dan menghasilkan tanda tangan individu yang dapat diverifikasi di lebih dari satu wilayah. AWS Dengan Sigv4a, Anda dapat menandatangani permintaan untuk beberapa wilayah, dengan perutean yang mulus dan failover antar wilayah. Saat Anda menggunakan AWS SDK atau AWS CLI untuk memanggil fungsionalitas yang memerlukan penandatanganan multi-wilayah, jenis tanda tangan secara otomatis diubah untuk menggunakan Sigv4a. Lihat perinciannya di Bagaimana AWS SigV4a berfungsi.

Bagaimana AWS SigV4 berfungsi

Langkah-langkah berikut menjelaskan proses umum menghitung tanda tangan dengan SIGv4:

1. String yang akan ditandatangani tergantung pada jenis permintaan. Misalnya, saat Anda menggunakan header Otorisasi HTTP atau parameter kueri untuk otentikasi, Anda menggunakan kombinasi elemen permintaan untuk membuat string yang akan ditandatangani. Untuk permintaan HTTP POST, `POST` kebijakan dalam permintaan adalah string yang Anda tandatangani.
2. K unci penandatanganan adalah serangkaian perhitungan, dengan hasil dari setiap langkah dimasukkan ke langkah berikutnya. Langkah terakhir adalah kunci penandatanganan.
3. Ketika AWS layanan menerima permintaan yang diautentikasi, ia membuat ulang tanda tangan menggunakan informasi otentikasi yang terkandung dalam permintaan. Jika tanda tangan cocok, layanan memproses permintaan. Jika tidak, ia menolak permintaan tersebut.

Untuk informasi selengkapnya, lihat Elemen dari sebuah AWS Tanda tangan permintaan API.

Bagaimana AWS SigV4a berfungsi

Sigv4a menggunakan tanda tangan asimetris berdasarkan kriptografi kunci publik-swasta. SigV4a melalui proses derivasi kredenSIAL cakupan yang serupa dengan SIGv4, kecuali SigV4a menggunakan kunci yang sama untuk menandatangani semua permintaan tanpa perlu mendapatkan kunci penandatanganan yang berbeda berdasarkan tanggal, layanan, dan wilayah. Keypair Elliptic Curve Digital Signature Alg orithm (ECDSA) dapat diturunkan dari kunci akses rahasia Anda yang ada AWS .

Sistem menggunakan kriptografi asimetris untuk memverifikasi tanda tangan multi-wilayah, sehingga AWS hanya perlu menyimpan kunci publik Anda. Kunci publik tidak rahasia dan tidak dapat digunakan untuk menandatangani permintaan. Tanda tangan asimetris diperlukan untuk permintaan API multi-wilayah, seperti dengan Amazon S3 Multi-Region Access Points.

Langkah-langkah berikut menjelaskan proses umum komputasi tanda tangan dengan Sigv4a:

1. String yang akan ditandatangani tergantung pada jenis permintaan. Misalnya, saat Anda menggunakan header Otorisasi HTTP atau parameter kueri untuk otentikasi, Anda menggunakan kombinasi elemen permintaan untuk membuat string yang akan ditandatangani. Untuk permintaan HTTP POST, `POST` kebijakan dalam permintaan adalah string yang Anda tandatangani.
2. K unci penandatanganan berasal dari kunci akses AWS rahasia melalui serangkaian perhitungan, dengan hasil dari setiap langkah dimasukkan ke langkah berikutnya. Langkah terakhir menghasilkan keypair.
3. Ketika AWS layanan menerima permintaan yang ditandatangani dengan SigV4a, AWS memverifikasi tanda tangan hanya menggunakan setengah publik dari pasangan kunci. Jika tanda tangan valid, permintaan diautentikasi dan layanan memproses permintaan. Permintaan dengan tanda tangan tidak valid ditolak.

Untuk informasi selengkapnya tentang Sigv4a untuk permintaan API Multi-wilayah, lihat proyek sigv4a- sig ning-examples di. GitHub

Kapan menandatangani permintaan

Saat Anda menulis kode khusus yang mengirimkan permintaan API ke AWS, Anda harus menyertakan kode yang menandatangani permintaan. Anda dapat menulis kode khusus karena:

- Anda bekerja dengan bahasa pemrograman yang tidak ada SDK AWS .
- Anda perlu kontrol penuh atas bagaimana permintaan dikirim ke AWS.

Sementara permintaan API mengautentikasi akses dengan AWS SIGv4, AWS SDK dan mengo AWS CLI tentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Untuk informasi selengkapnya tentang otentikasi dengan AWS SDK dan AWS CLI, lihat. Sumber daya tambahan

Mengapa permintaan ditandatangani

Proses penandatanganan membantu mengamankan permintaan dengan cara berikut:

- Verifikasi identitas pemohon

  Permintaan yang diautentikasi memerlukan tanda tangan yang Anda buat dengan menggunakan kunci akses Anda (ID kunci akses, kunci akses rahasia). Jika Anda menggunakan kredentif keamanan sementara, perhitungan tanda tangan juga memerlukan token keamanan. Untuk informasi selengkapnya, lihat akses terprogram kredensional AWS keamanan.
- Lindungi data saat transit

  Untuk mencegah gangguan pada permintaan saat sedang transit, beberapa elemen permintaan digunakan untuk menghitung hash (intisari) permintaan, dan nilai hash yang dihasilkan disertakan sebagai bagian dari permintaan. Ketika Layanan AWS menerima permintaan, ia menggunakan informasi yang sama untuk menghitung hash dan mencocokkannya dengan nilai hash dalam permintaan Anda. Jika nilainya tidak cocok, AWS tolak permintaan.
- Melindungi dari potensi serangan replay

  Dalam kebanyakan kasus, permintaan harus mencapai AWS dalam waktu lima menit dari cap waktu dalam permintaan. Jika tidak, AWS menolak permintaan tersebut.

AWS SIGv4 dapat dinyatakan dalam header Otorisasi HTTP atau sebagai string kueri di URL. Untuk informasi selengkapnya, lihat Metode autentikasi.

Sumber daya tambahan

- Untuk informasi selengkapnya tentang proses penandatanganan SIGv4 untuk berbagai layanan, lihatMinta contoh tanda tangan.

- Untuk mengonfigurasi kredenSIAL untuk akses terprogram untuk AWS CLI, lihat Oten tikasi dan kredenSIAL akses di Panduan Pengguna Antar AWS muka Baris Perintah.
- SDK AWS menyertakan kode sumber GitHub untuk menandatangani permintaan AWS API. Untuk contoh kode, lihatContoh proyek di AWS repositori sampel.
  - AWS SDK untuk .NET – AWSSigV4Signer.cs
  - AWS SDK untuk C++ – AWSAuthV4Signer.cpp
  - AWS SDK untuk Go - sigv4.go
  - AWS SDK untuk Java – BaseAws4Signer.java
  - AWS SDK untuk JavaScript — tanda tang an-v4
  - AWS SDK untuk PHP – SignatureV4.php
  - AWS SDK for Python (Boto) - signers.py
  - AWS SDK untuk Ruby — signatur.rb
- Jika Anda mengalami kesalahan saat menandatangani permintaan, lihatMemecahkan masalah penandatanganan Tanda Tangan Versi 4 untuk AWS Permintaan API.

Elemen dari sebuah AWS Tanda tangan permintaan API

> **Important**
>
> Kecuali Anda menggunakan AWS SDK atau CLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan AWS tanda tangan di Tanda Tangan Versi 4 bisa menjadi tugas yang rumit, dan kami menyarankan Anda menggunakan AWS SDK atau CLI bila memungkinkan.

Setiap HTTP/HTTPS permintaan yang menggunakan penandatanganan Signature Version 4 harus berisi elemen-elemen ini.

Elemen

- Spesifikasi titik akhir
- Tindakan
- Parameter tindakan
- Date
- Informasi otentikasi

Spesifikasi titik akhir

Menentukan nama DNS titik akhir tempat Anda mengirim permintaan. Nama ini biasanya berisi kode layanan dan Wilayah. Misalnya, titik akhir untuk Amazon DynamoDB di Wil `us-east-1` ayah adalah. `dynamodb.us-east-1.amazonaws.com`

Untuk HTTP/1.1 permintaan, Anda harus menyertakan `Host` header. Untuk HTTP/2 permintaan, Anda dapat menyertakan `:authority` header atau `Host` header. Gunakan hanya `:authority` header untuk kepatuhan dengan HTTP/2 spesifikasi. Tidak semua layanan mendukung HTTP/2 permintaan.

Untuk titik akhir yang didukung oleh setiap layanan, lihat Titik akhir layanan dan kuota di. Referensi Umum AWS

Tindakan

Menentukan tindakan API untuk layanan. Misalnya, tindakan DynamoDB atau `CreateTable` tindakan Amazon EC `DescribeInstances` 2.

Untuk tindakan yang didukung oleh setiap layanan, lihat Referensi O torisasi Layanan.

Parameter tindakan

Menentukan parameter untuk tindakan yang ditentukan dalam permintaan. Setiap tindakan AWS API memiliki satu set parameter wajib dan opsional. Versi API biasanya merupakan parameter yang diperlukan.

Untuk parameter yang didukung oleh tindakan API, lihat Referensi API untuk layanan.

Date

Menentukan tanggal dan waktu permintaan. Menyertakan tanggal dan waktu dalam permintaan membantu mencegah pihak ketiga mencegat permintaan Anda dan mengirimkannya kembali nanti. Tanggal yang Anda tentukan dalam cakupan kredensi harus sesuai dengan tanggal permintaan Anda.

Cap waktu harus dalam UTC dan menggunakan format ISO 8601 berikut: YYYYMMDD T HHMMSS Z. Misalnya, `20220830T123600Z`. Jangan sertakan milidetik dalam stempel waktu.

Anda dapat menggunakan a `date` atau `x-amz-date` header, atau menyertakan `x-amz-date` sebagai parameter kueri. Jika kita tidak dapat menemukan `x-amz-date` header, maka kita mencari `date` header.

Informasi otentikasi

Setiap permintaan yang Anda kirim harus menyertakan informasi berikut. AWS menggunakan informasi ini untuk memastikan validitas dan keaslian permintaan.

- Algoritma — Algoritma yang Anda gunakan sebagai bagian dari proses penandatanganan.
  - SIGv4 - Gunakan `AWS4-HMAC-SHA256` untuk menentukan Signature Version 4 dengan algoritma `HMAC-SHA256` hash.
  - Sigv4a — Gunakan `AWS4-ECDSA-P256-SHA256` untuk menentukan algoritma `ECDSA-P256-SHA-256` hash.
- Credential — Sebuah string yang dibentuk dengan menggabungkan ID kunci akses Anda dan komponen cakupan kredensia Anda.
  - SIGv4 — Cakupan kredensi mencakup ID kunci akses Anda, tanggal dalam format YYYYMMDD, kode Wilayah, kode layanan, dan string `aws4_request` terminasi, dipisahkan oleh garis miring (/). Kode Wilayah, kode layanan, dan string terminasi harus menggunakan karakter huruf kecil.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
    ```

  - SigV4a — Cakupan kredensi mencakup tanggal dalam format YYYYMMDD, nama layanan, dan string `aws4_request` terminasi, dipisahkan oleh garis miring (/). Perhatikan bahwa cakupan kredenSIAL tidak menyertakan wilayah karena wilayah tersebut tercakup dalam header `X-Amz-Region-Set` terpisah.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/service/aws4_request
    ```

- Header yang ditandatangani — Header HTTP untuk disertakan dalam tanda tangan, dipisahkan oleh titik koma (;). Misalnya, `host;x-amz-date`.

  Untuk SigV4a, Anda harus menyertakan header set wilayah yang menentukan kumpulan wilayah tempat permintaan akan valid. Header `X-Amz-Region-Set` ditentukan sebagai daftar nilai yang dipisahkan koma. Contoh berikut menunjukkan header wilayah yang memungkinkan permintaan dibuat di wilayah us-east-1 dan us-west-1.

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  Anda dapat menggunakan wildcard (*) di wilayah untuk menentukan beberapa wilayah. Dalam contoh berikut, header memungkinkan permintaan dibuat di us-west-1 dan us-west-2.

```
X-Amz-Region-Set=us-west-*
```

- Signature — Sebuah string berkode heksadesimal yang mewakili tanda tangan yang dihitung. Anda harus menghitung tanda tangan menggunakan algoritme yang Anda tentukan di parameter `Algorithm`.

Untuk informasi selengkapnya, lihat Metode autentikasi

Metode autentikasi

> **Important**
>
> Kecuali Anda menggunakan AWS SDK atau CLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan AWS tanda tangan di Tanda Tangan Versi 4 bisa menjadi tugas yang rumit, dan kami menyarankan Anda menggunakan AWS SDK atau CLI bila memungkinkan.

Anda dapat mengungkapkan informasi otentikasi dengan menggunakan salah satu metode berikut.

Header otorisasi HTTP

`Authorization`Header HTTP adalah metode yang paling umum untuk mengautentikasi permintaan. Semua operasi REST API (kecuali untuk unggahan berbasis browser menggunakan `POST` permintaan) memerlukan header ini.

Contoh berikut menunjukkan nilai `Authorization` header untuk SIGv4 dan SigV4a. Istirahat baris ditambahkan ke contoh ini untuk keterbacaan. Dalam kode Anda, header harus berupa string kontinu. Tidak ada koma antara algoritma dan Credential, tetapi elemen lainnya harus dipisahkan dengan koma.

Example SigV4

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Example SigV4A

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/s3/aws4_request,
SignedHeaders=host;range;x-amz-date;x-amz-region-set,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Tabel berikut menjelaskan berbagai komponen dari nilai header Otorisasi dalam contoh sebelumnya:

Komponen	Deskripsi
Otorisasi	Algoritma yang digunakan untuk menghitung tanda tangan. • SigV4 - Gunakan`AWS4-HMAC-SHA256` . String ini mengidentifikasi AWS sigv4 (`AWS4`) dan `HMAC-SHA256` algoritma. • Sigv4a — Gunakan. `AWS4-ECDSA-P256-SHA256` String ini mengidentifikasi AWS sigv4 (`AWS4`) dan `ECDSA-P256-SHA-256` algoritma.

Kredensimen	ID kunci akses Anda dan informasi ruang lingkup.
	• SIGv4 — Sertakan tanggal, Wilayah, dan layanan yang digunakan untuk menghitung tanda tangan. String ini memiliki bentuk berikut:
	`<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request`
	• Sigv4a — Sertakan tanggal dan layanan yang digunakan untuk menghitung tanda tangan. String ini memiliki bentuk berikut:
	`<your-access-key-id>/<date>/ <aws-service>/aws4_request`

Komponen	Deskripsi
	<date>Nilai ditentukan menggunakan format YYYYMMDD. <aws-service>nilainya adalah S3 saat mengirim permintaan ke Amazon S3.
SignedHeaders	Daftar header permintaan yang dipisahkan titik koma yang Anda gunakan untuk menghitun g Tanda Tangan. Daftar ini hanya mencakup nama header, dan nama header harus dalam huruf kecil. Misalnya: `host;range;x-amz-date` Untuk SigV4a, Anda harus menyertakan header set wilayah yang menentukan kumpulan wilayah tempat permintaan akan valid. Header X-Amz-Region-Set ditentukan sebagai daftar nilai yang dipisahkan koma.
Tanda tangan	Tanda tangan 256-bit dinyatakan sebagai 64 karakter heksadesimal huruf kecil. Sebagai contoh: `fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024` Perhatikan bahwa perhitungan tanda tangan bervariasi tergantung pada opsi yang Anda pilih untuk mentransfer payload.

Parameter string kueri

Anda dapat menggunakan string kueri untuk mengekspresikan permintaan sepenuhnya dalam URL. Dalam hal ini, Anda menggunakan parameter kueri untuk memberikan informasi permintaan, termasuk informasi otentikasi. Karena tanda tangan permintaan adalah bagian dari URL, jenis URL ini sering disebut sebagai URL yang ditandatangani sebelumnya. Anda dapat menggunakan URL yang telah ditandatangani sebelumnya untuk menyematkan tautan yang dapat diklik dalam HTML, yang dapat berlaku hingga tujuh hari. Untuk informasi selengkapnya, lihat Mengautentikasi Permintaan: Menggunakan Parameter Kueri AWS (Versi Tangan 4) di Refer ensi API Amazon S3.

Contoh berikut menunjukkan URL yang telah ditandatangani sebelumnya untuk SIGv4 dan Sigv4a. Istirahat baris ditambahkan ke contoh ini untuk keterbacaan:

Example SigV4

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/<region>/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

Example SigV4A

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256 &
X-Amz-Credential=<your-access-key-id>/20240721/s3/aws4_request &
X-amz-Region-Set=<regionset> &
X-Amz-Date=20240721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host;x-amz-region-set &
X-Amz-Signature=<signature-value>
```

> **Note**
>
> `X-Amz-Credential`Nilai dalam URL menunjukkan karakter “/” hanya untuk keterbacaan. Dalam praktiknya, itu harus dikodekan sebagai %2F. Contoh: `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

Tabel berikut menjelaskan parameter kueri di URL yang menyediakan informasi otentikasi.

Nama parameter string kueri	Deskripsi
X-Amz-Algorithm	Versi AWS tanda tangan dan algoritma yang Anda gunakan untuk menghitung tanda tangan.

Nama parameter string kueri	Deskripsi
	• SigV4 - Gunakan`AWS4-HMAC-SHA256` . String ini mengidentifikasi AWS sigv4 (`AWS4`) dan `HMAC-SHA256` algoritma. • Sigv4a — Gunakan. `AWS4-ECDSA-P256-SHA256` String ini mengidentifikasi AWS sigv4 (`AWS4`) dan `ECDSA-P256-SHA-256` algoritma.

Nama parameter string kueri	Deskripsi
X-Amz-Credential	Selain ID kunci akses Anda, parameter ini juga menyediakan ruang lingkup yang valid tanda tangan. Nilai ini harus sesuai dengan cakupan yang Anda gunakan dalam perhitungan tanda tangan, dibahas di bagian berikut. • SigV4 — Bentuk umum untuk nilai parameter ini adalah sebagai berikut: `<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request` Misalnya: `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` • Sigv4a — Bentuk umum untuk nilai parameter ini adalah sebagai berikut: `<your-access-key-id>/<date>/<AWS-service>/aws4_request` Misalnya: `AKIAIOSFODNN7EXAMPLE/20130721/s3/aws4_request` Wilayah untuk SigV4a didefinisikan dalam header set wilayah. `X-Amz-Region-Set` Untuk daftar string AWS regional, lihat Titik Akhir Regional di Referensi AWS Umum.
X-Amz-Region-Set	Kumpulan wilayah tempat permintaan akan valid. Header x-amz-region-set ditentukan sebagai daftar nilai yang dipisahkan koma.

Nama parameter string kueri	Deskripsi
X-Amz-Date	Format tanggal dan waktu harus mengikuti standar ISO 8601, dan harus diformat dengan `yyyyMMddTHHmmssZ` format. Misalnya jika tanggal dan waktu adalah "08/01/2016 15:32:41.982-700" maka pertama-tama harus dikonversi ke UTC (Waktu Universal Terkoordinasi) dan kemudian dikirimkan sebagai “20160801T223241Z”.
X-Amz-Expires	Menyediakan periode waktu, dalam detik, di mana URL yang telah ditandatangani sebelumnya valid. Misalnya, 86400 (24 jam). Nilai ini adalah bilangan bulat. Nilai minimum yang dapat Anda tetapkan adalah 1, dan maksimum adalah 604800 (tujuh hari). URL yang telah ditandatangani dapat berlaku maksimal tujuh hari karena kunci penandatanganan yang Anda gunakan dalam perhitungan tanda tangan berlaku hingga tujuh hari.

Nama parameter string kueri	Deskripsi
X-Amz-SignedHeaders	Mencantumkan header yang Anda gunakan untuk menghitung tanda tangan. Header berikut diperlukan dalam perhitungan tanda tangan: • Header host HTTP. • Setiap header x-amz-* yang Anda rencanakan untuk ditambahkan ke permintaan. • Untuk Sigv4a, X-Amz-Region-Set diperlukan untuk menentukan wilayah di mana permintaan dapat dibuat. Untuk keamanan tambahan, Anda harus menandatangani semua header permintaan yang Anda rencanakan untuk disertakan dalam permintaan Anda.
X-Amz-Signature	Menyediakan tanda tangan untuk mengotentikasi permintaan Anda. Tanda tangan ini harus sesuai dengan tanda tangan yang dihitung layanan; jika tidak, layanan menolak permintaan. Sebagai contoh, `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` . Perhitungan tanda tangan dijelaskan di bagian berikut.
X-Amz-Security-Token	Parameter kredensi opsional jika menggunakan kredenSIAL yang bersumber dari layanan STS.

Buat yang ditandatangani AWS Permintaan API

> **Important**
>
> Jika Anda menggunakan alat AWS SDK (lihat Kode Sam pel dan Pustaka AWS CLI) atau AWS Command Line Interface () untuk mengirim permintaan API AWS, Anda dapat melewati bagian ini karena klien SDK dan CLI mengotentikasi permintaan Anda dengan menggunakan kunci akses yang Anda berikan. Kecuali Anda memiliki alasan bagus untuk tidak melakukannya, kami sarankan Anda selalu menggunakan SDK atau CLI.
> Di Wilayah yang mendukung beberapa versi tanda tangan, permintaan penandatanganan secara manual berarti Anda harus menentukan versi tanda tangan mana yang digunakan. Saat Anda memberikan permintaan ke Multi-Region Access Points, SDK dan CLI secara otomatis beralih menggunakan Signature Version 4A tanpa konfigurasi tambahan.

Anda dapat menggunakan protokol penandatanganan AWS SIGv4 untuk membuat permintaan yang ditandatangani untuk permintaan AWS API.

1. Membuat permintaan kanonik berdasarkan detail permintaan.
2. Menghitung tanda tangan menggunakan AWS kredensibilitas Anda.
3. Menambahkan tanda tangan ini ke permintaan sebagai header Otorisasi.

AWS kemudian mereplikasi proses ini dan memverifikasi tanda tangan, memberikan atau menolak akses yang sesuai.

Untuk melihat bagaimana Anda dapat menggunakan AWS SIGv4 untuk menandatangani permintaan API, lihatMinta contoh tanda tangan.

Tabel berikut menjelaskan fungsi yang digunakan dalam proses membuat permintaan yang ditandatangani. Anda perlu menerapkan kode untuk fungsi-fungsi ini. Untuk informasi selengkapnya, lihat contoh kode di AWS SDK.

Fungsi	Deskripsi
`Lowercase()`	Ubah string menjadi huruf kecil.
`Hex()`	Pengkodean huruf kecil 16.

Fungsi	Deskripsi
`SHA256Hash()`	Fungsi hash kriptografi Secure Hash Algorithm (SHA).
`HMAC-SHA256()`	Menghitung HMAC dengan menggunakan algoritma SHA256 dengan kunci penandata nganan yang disediakan. Ini adalah tanda tangan terakhir saat Anda menandatangani dengan SIGv4.
`ECDSA-Sign`	Tanda tangan Elliptic Curve Digital Signature Algorithm (ECDSA) dihitung dengan menggunakan tanda tangan asimetris berdasarkan kriptografi kunci publik-swasta.
`KDF(K, Label, Context, L)`	Sebuah NIST SP800-108 KDF dalam Mode Penghitung menggunakan fungsi PRF seperti yang didefinisikan dalam NIST HMAC-SHA256 SP 800-108r1.
`Oct2Int(byte[ ])`	Fungsi oktet ke bilangan bulat seperti yang dijelaskan dalam ANSI X9.62.
`Trim()`	Hapus spasi kosong di depan atau di belakang.

Fungsi	Deskripsi
`UriEncode()`	URI mengkodekan setiap byte. UriEncode() harus menegakkan aturan berikut: • URI mengkodekan setiap byte kecuali karakter tanpa syarat: 'A'-'Z', 'a'-'z', '0'-'9', '-', ' . ',' _ ', dan '~'. • Karakter spasi adalah karakter yang dicadangkan dan harus dikodekan sebagai “%20" (dan bukan sebagai “+”). • Setiap byte yang dikodekan URI dibentuk oleh '%' dan nilai heksadesimal dua digit byte. • Huruf dalam nilai heksadesimal harus huruf besar, misalnya “%1A”. • Enkode karakter garis miring ke depan, '/', di mana-mana kecuali dalam nama kunci objek. Misalnya, jika nama kunci objek adalah`photos/Jan/sample.jpg` , garis miring ke depan dalam nama kunci tidak dikodekan. **Important** UriEncode Fungsi standar yang disediakan oleh platform pengemban gan Anda mungkin tidak berfungsi karena perbedaan dalam implementasi dan ambiguitas terkait dalam RFC yang mendasarinya. Kami menyarankan Anda menulis UriEncode fungsi kustom Anda sendiri untuk memastikan bahwa pengkodean Anda akan berfungsi.

Fungsi	Deskripsi
	Untuk melihat contoh UriEncode fungsi di Java, lihat Utilitas Java di GitHub situs web.

> **Note**
>
> Saat menandatangani permintaan Anda, Anda dapat menggunakan AWS SIGv4 atau AWS SIGv4a. Perbedaan utama antara keduanya ditentukan oleh bagaimana tanda tangan dihitung. Dengan SigV4a, set wilayah disertakan dalam string yang akan ditandatangani, tetapi bukan bagian dari langkah derivasi kredensia.

Menandatangani permintaan dengan kredentif keamanan sementara

Alih-alih menggunakan kredenSIAL jangka panjang untuk menandatangani permintaan, Anda dapat menggunakan kredenSIAL keamanan sementara yang disediakan oleh AWS Security Token Service (AWS STS).

Saat Anda menggunakan kredentif keamanan sementara, Anda harus menambahkan `X-Amz-Security-Token` ke header Otorisasi atau memasukkannya ke dalam string kueri untuk menyimpan token sesi. Beberapa layanan mengharuskan Anda `X-Amz-Security-Token` menambahkan permintaan kanonik. Layanan lain hanya mengharuskan Anda menambahkan `X-Amz-Security-Token` di akhir, setelah Anda menghitung tanda tangan. Periksa dokumentasi untuk masing-masing Layanan AWS untuk persyaratan tertentu.

Ringkasan langkah-langkah penandatanganan

Buat permintaan kanonik

Atur isi permintaan Anda (host, tindakan, header, dll.) Ke dalam format kanonik standar. Permintaan kanonik adalah salah satu input yang digunakan untuk membuat string untuk ditandatangani. Untuk detail tentang membuat permintaan kanonik, lihatElemen dari sebuah AWS Tanda tangan permintaan API.

Buat hash dari permintaan kanonik

Hash permintaan kanonik menggunakan algoritma yang sama yang Anda gunakan untuk membuat hash payload. Hash permintaan kanonik adalah string karakter heksadesimal huruf kecil.

Buat string untuk ditandatangani

Buat string untuk ditandatangani dengan permintaan kanonik dan informasi tambahan seperti algoritma, tanggal permintaan, cakupan kredenSIAL, dan hash permintaan kanonik.

Dapatkan kunci penandatanganan

Gunakan kunci akses rahasia untuk mendapatkan kunci yang digunakan untuk menandatangani permintaan.

Hitung tanda tangan

Lakukan operasi hash dengan kunci pada string untuk menandatangani menggunakan kunci penandatanganan turunan sebagai kunci hash.

Tambahkan tanda tangan ke permintaan

Tambahkan tanda tangan yang dihitung ke header HTTP atau ke string kueri permintaan.

Buat permintaan kanonik

Untuk membuat permintaan kanonik, gabungkan string berikut, dipisahkan oleh karakter baris baru. Ini membantu memastikan bahwa tanda tangan yang Anda hitung dapat cocok dengan tanda tangan yang di AWS hitung.

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod*— Metode HTTP, sepertiGET,PUT,HEAD, danDELETE.
- *CanonicalUri*— URI-encoded Versi URI komponen jalur absolut, dimulai dengan / yang mengikuti nama domain dan hingga akhir string atau ke karakter tanda tanya (?) jika Anda memiliki parameter string kueri. Jika jalur absolut kosong, gunakan karakter garis miring maju (/). URI dalam contoh berikut,/amzn-s3-demo-bucket/myphoto.jpg, adalah jalur absolut dan Anda tidak meny / andikan jalur absolut:

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.jpg
  ```

- *CanonicalQueryString*— Param URI-encoded eter string kueri. Anda URI-encode masing-masing nama dan nilai secara individual. Anda juga harus mengurutkan parameter dalam string kueri kanonik menurut abjad berdasarkan nama kunci. Penyortiran terjadi setelah pengkodean. String kueri dalam contoh URI berikut adalah:

```
http://s3.amazonaws.com/amzn-s3-demo-bucket?prefix=somePrefix&marker=someMarker&max-keys=2
```

String kueri kanonik adalah sebagai berikut (jeda baris ditambahkan ke contoh ini untuk keterbacaan):

```
UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
UriEncode("max-keys")+"="+UriEncode("20") + "&" +
UriEncode("prefix")+"="+UriEncode("somePrefix")
```

Ketika permintaan menargetkan subresource, nilai parameter query yang sesuai akan menjadi string kosong (""). Misalnya, URI berikut mengidentifikasi `ACL` subresource pada `amzn-s3-demo-bucket` bucket:

```
http://s3.amazonaws.com/amzn-s3-demo-bucket?acl
```

Dalam hal ini, itu CanonicalQueryString akan menjadi:

```
UriEncode("acl") + "=" + ""
```

Jika URI tidak menyertakan a?, tidak ada string kueri dalam permintaan, dan Anda mengatur string kueri kanonik ke string kosong (""). Anda masih perlu memasukkan karakter baris baru ("\n").

- *CanonicalHeaders*— Daftar header permintaan dengan nilainya. Nama header individu dan pasangan nilai dipisahkan oleh karakter baris baru ("\n"). Berikut ini adalah contoh dari CanonicalHeader:

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders daftar harus mencakup yang berikut:

- `host`Header HTTP.
- Jika `Content-Type` header ada dalam permintaan, Anda harus menambahkannya ke *`CanonicalHeaders`* daftar.
- Setiap `x-amz-*` header yang Anda rencanakan untuk disertakan dalam permintaan Anda juga harus ditambahkan. Misalnya, jika Anda menggunakan kredentif keamanan sementara, Anda harus menyertakan `x-amz-security-token` dalam permintaan Anda. Anda harus menambahkan header ini dalam daftar*`CanonicalHeaders`*.
- Untuk SigV4a, Anda harus menyertakan header set wilayah yang menentukan kumpulan wilayah tempat permintaan akan valid. Header `X-Amz-Region-Set` ditentukan sebagai daftar nilai yang dipisahkan koma. Contoh berikut menunjukkan header wilayah yang memungkinkan permintaan dibuat di wilayah us-east-1 dan us-west-1.

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  Anda dapat menggunakan wildcard (*) di wilayah untuk menentukan beberapa wilayah. Dalam contoh berikut, header memungkinkan permintaan dibuat di us-west-1 dan us-west-2.

  ```
  X-Amz-Region-Set=us-west-*
  ```

> **Note**
>
> `x-amz-content-sha256`Header diperlukan untuk AWS permintaan Amazon S3. Ini menyediakan hash dari payload permintaan. Jika tidak ada payload, Anda harus memberikan hash dari string kosong.

Setiap nama header harus:

- gunakan karakter huruf kecil.
- muncul dalam urutan abjad.
- diikuti oleh titik dua (`:`).

Untuk nilai, Anda harus:

- potong spasi depan atau belakang.
- mengubah spasi berurutan menjadi satu ruang.

- pisahkan nilai untuk header multi-nilai menggunakan koma.
- Anda harus menyertakan header host (HTTP/1.1) atau:otoritas header (HTTP/2), dan `x-amz-*` header apa pun dalam tanda tangan. Anda dapat secara opsional menyertakan header standar lainnya dalam tanda tangan, seperti tipe konten.

`Trim()`Fungsi `Lowercase()` dan yang digunakan dalam contoh ini dijelaskan di bagian sebelumnya.

Berikut ini adalah contoh `CanonicalHeaders` string. Nama header dalam huruf kecil dan diurutkan.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Untuk tujuan menghitung tanda tangan otorisasi, hanya host dan `x-amz-*` header apa pun yang diperlukan; Namun, untuk mencegah gangguan data, Anda harus mempertimbangkan untuk memasukkan header tambahan dalam perhitungan tanda tangan.
> Jangan sertakan header hop-by-hop yang sering diubah selama transit melintasi sistem yang kompleks. Ini termasuk semua header transport volatile yang dimutasi oleh proxy, penyeimbang beban, dan node dalam sistem terdistribusi, termasuk,,`connection`,,`x-amzn-trace-id`,`user-agent`,`keep-alive`, `transfer-encoding` `TE``trailer`, `upgrade` dan. `proxy-authorization` `proxy-authenticate`

- *`SignedHeaders`*— Daftar nama header permintaan huruf kecil yang diurutkan berdasarkan abjad dan dipisahkan titik koma. Header permintaan dalam daftar adalah header yang sama yang Anda sertakan dalam string. `CanonicalHeaders` Untuk contoh sebelumnya, nilainya *`SignedHeaders`* adalah sebagai berikut:

```
host;x-amz-content-sha256;x-amz-date
```

- *`HashedPayload`*— Sebuah string yang dibuat menggunakan payload di badan permintaan HTTP sebagai input ke fungsi hash. String ini menggunakan karakter heksadesimal huruf kecil.

```
Hex(SHA256Hash(<payload>))
```

Jika tidak ada payload dalam permintaan, Anda menghitung hash dari string kosong, seperti ketika Anda mengambil objek dengan menggunakan `GET` permintaan, tidak ada apa pun dalam payload.

```
Hex(SHA256Hash(""))
```

> **Note**
>
> Untuk Amazon S3, sertakan string literal `UNSIGNED-PAYLOAD` saat membuat permintaan kanonik, dan tetapkan nilai yang sama dengan nilai `x-amz-content-sha256` header saat mengirim permintaan.
> `Hex(SHA256Hash("UNSIGNED-PAYLOAD"))`

Buat hash permintaan kanonik

Buat hash (ringkasan) permintaan kanonik menggunakan algoritma yang sama yang Anda gunakan untuk membuat hash payload. Hash permintaan kanonik adalah string karakter heksadesimal huruf kecil.

Buat string untuk ditandatangani

Untuk membuat string untuk ditandatangani, gabungkan string berikut, dipisahkan oleh karakter baris baru. Jangan mengakhiri string ini dengan karakter baris baru.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *`Algorithm`*Algoritma yang digunakan untuk membuat hash permintaan kanonik.
  - SigV4 — Digunakan `AWS4-HMAC-SHA256` untuk menentukan algoritma `HMAC-SHA256` hash.
  - Sigv4a — Gunakan `AWS4-ECDSA-P256-SHA256` untuk menentukan algoritma `ECDSA-P256-SHA-256` hash.
- *`RequestDateTime`*— Tanggal dan waktu yang digunakan dalam lingkup kredenSIAL. Nilai ini adalah waktu UTC saat ini dalam format ISO 8601 (misalnya,`20130524T000000Z`).

- *CredentialScope*— Lingkup kredenSIAL, yang membatasi tanda tangan yang dihasilkan ke Wilayah dan layanan yang ditentukan.
  - SigV4 - Kredensi termasuk ID kunci akses Anda, tanggal dalam YYYYMMDD format, kode Wilayah, kode layanan, dan string `aws4_request` penghentian, dipisahkan oleh garis miring (/). Kode Wilayah, kode layanan, dan string terminasi harus menggunakan karakter huruf kecil. String memiliki format berikut:YYYYMMDD/`region/service/aws4_request`.
  - SigV4a - KredenSIAL mencakup tanggal dalam YYYYMMDD format, nama layanan, dan string `aws4_request` terminasi, dipisahkan oleh garis miring (/). Perhatikan bahwa cakupan kredenSIAL tidak menyertakan wilayah karena wilayah tersebut tercakup dalam header `X-Amz-Region-Set` terpisah. String memiliki format berikut:YYYYMMDD/`service/aws4_request`.
- *HashedCanonicalRequest*— Hash permintaan kanonik, dihitung pada langkah sebelumnya.

Berikut ini adalah contoh string untuk ditandatangani.

```
"<Algorithm>" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(<Algorithm>(<CanonicalRequest>))
```

Dapatkan kunci penandatanganan

Untuk mendapatkan kunci penandatanganan, pilih salah satu proses berikut untuk menghitung kunci penandatanganan untuk SIGv4 atau SIGV4a.

Memperoleh kunci penandatanganan untuk SIGv4

Untuk mendapatkan kunci penandatanganan untuk SIGv4, lakukan suksesi operasi hash kunci (HMAC) pada tanggal permintaan, Wilayah, dan layanan, dengan kunci akses AWS rahasia Anda sebagai kunci untuk operasi hashing awal.

Untuk setiap langkah, panggil fungsi hash dengan kunci dan data yang diperlukan. Hasil dari setiap panggilan ke fungsi hash menjadi input untuk panggilan berikutnya ke fungsi hash.

Contoh berikut menunjukkan bagaimana Anda memperoleh yang `SigningKey` digunakan di bagian selanjutnya dari prosedur ini, menunjukkan urutan di mana input Anda digabungkan dan di-hash. `HMAC-SHA256`adalah fungsi hash yang digunakan untuk hash data seperti yang ditunjukkan.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
```

```
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Masukan yang diperlukan

- `Key`— String yang berisi kunci akses rahasia Anda.
- `Date`— String yang berisi tanggal yang digunakan dalam lingkup kredenSIAL, dalam format YYYY MMDD.
- `Region`— Sebuah string yang berisi kode Wilayah (misalnya,`us-east-1`).

  Untuk daftar string Wilayah, lihat Titik Akhir Regional di Referensi Umum AWS.
- `Service`— Sebuah string yang berisi kode layanan (misalnya,`ec2`).
- String untuk menandatangani yang Anda buat di langkah sebelumnya.

Untuk mendapatkan kunci penandatanganan untuk SIGv4

1. Gabungkan `"AWS4"` dan kunci akses rahasia. Panggil fungsi hash dengan string gabungan sebagai kunci dan string tanggal sebagai data.

   ```
   DateKey = hash("AWS4" + Key, Date)
   ```
2. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string Region sebagai data.

   ```
   DateRegionKey = hash(kDate, Region)
   ```
3. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string layanan sebagai data.

   Kode layanan ditentukan oleh layanan. Anda dapat menggunakan get-products di Pr AWS icing CLI untuk mengembalikan kode layanan untuk layanan.

   ```
   DateRegionServiceKey = hash(kRegion, Service)
   ```
4. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan “aws4_request” sebagai data.

   ```
   SigningKey = hash(kService, "aws4_request")
   ```

Memperoleh kunci penandatanganan untuk Sigv4a

Untuk membuat kunci penandatanganan untuk Sigv4a, gunakan proses berikut untuk mendapatkan pasangan kunci dari kunci akses rahasia. Untuk contoh implementasi derivasi ini, lihat implementasi pustaka C99 dari AWS otentikasi sisi klien

```
n = [NIST P-256 elliptic curve group order]
G = [NIST P-256 elliptic curve base point]
label = "AWS4-ECDSA-P256-SHA256"

akid = [AWS access key ID as a UTF8 string]
sk = [AWS secret access Key as a UTF8 Base64 string]

input_key = "AWS4A" || sk
count = 1
while (counter != 255) {
  context = akid || counter // note: counter is one byte
  key = KDF(input_key, label, context, 256)
  c = Oct2Int(key)
  if (c > n - 2) {
    counter++
  } else {
    k = c + 1   // private key
    Q = k * G   // public key
  }
}

if (c < 255) {
  return [k, Q]
} else {
  return FAILURE
}
```

Hitung tanda tangan

Setelah Anda memperoleh kunci penandatanganan, hitung tanda tangan untuk ditambahkan ke permintaan Anda. Prosedur ini bervariasi berdasarkan versi tanda tangan yang Anda gunakan.

Untuk menghitung tanda tangan untuk SIGv4

1. Panggil fungsi hash dengan hasil panggilan sebelumnya sebagai kunci dan string untuk ditandatangani sebagai data. Gunakan kunci penandatanganan turunan sebagai kunci hash untuk operasi ini. Hasilnya adalah tanda tangan sebagai nilai biner.

   ```
   signature = hash(SigningKey, string-to-sign)
   ```

2. Ubah tanda tangan dari biner ke representasi heksadesimal, dalam karakter huruf kecil.

Untuk menghitung tanda tangan untuk Sigv4a

1. Menggunakan algoritma penandatanganan digital (ECDSA P-256), tandatangani string untuk menandatangani yang Anda buat pada langkah sebelumnya. Kunci yang digunakan untuk tanda tangan ini adalah kunci asimetris pribadi yang berasal dari kunci akses rahasia seperti yang dijelaskan di atas.

   ```
   signature = base16(ECDSA-Sign(k, string-to-sign))
   ```

2. Ubah tanda tangan dari biner ke representasi heksadesimal, dalam karakter huruf kecil.

Tambahkan tanda tangan ke permintaan

Tambahkan tanda tangan yang dihitung ke permintaan Anda.

Example Contoh: Header otorisasi

SigV4

Contoh berikut menunjukkan `Authorization` header untuk `DescribeInstances` tindakan menggunakan AWS SIGv4. Untuk keterbacaan, contoh ini diformat dengan jeda baris. Dalam kode Anda, ini harus berupa string kontinu. Tidak ada koma antara algoritme dan `Credential`. Namun, elemen lainnya harus dipisahkan dengan koma.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

SiGV4A

Contoh berikut menunjukkan header Otorisasi untuk `CreateBucket` tindakan menggunakan AWS Sigv4a. Untuk keterbacaan, contoh ini diformat dengan jeda baris. Dalam kode Anda, ini harus berupa string kontinu. Tidak ada koma antara algoritma dan Credential. Namun, elemen lainnya harus dipisahkan dengan koma.

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-region-set,
Signature=calculated-signature
```

Example Contoh: Permintaan dengan parameter otentikasi dalam string kueri

SigV4

Contoh berikut menunjukkan kueri untuk `DescribeInstances` tindakan menggunakan AWS SIGv4 yang menyertakan informasi otentikasi. Untuk keterbacaan, contoh ini diformat dengan jeda baris dan tidak dikodekan URL. Dalam kode Anda, string kueri harus berupa string kontinu yang dikodekan URL.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

SiGV4A

Contoh berikut menunjukkan kueri untuk `CreateBucket` tindakan menggunakan AWS Sigv4a yang menyertakan informasi otentikasi. Untuk keterbacaan, contoh ini diformat dengan jeda baris dan tidak dikodekan URL. Dalam kode Anda, string kueri harus berupa string kontinu yang dikodekan URL.

```
https://ec2.amazonaws.com/?
Action=CreateBucket&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request&
X-Amz-Region-Set=us-west-1&
```

```
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date;x-amz-region-set&
X-Amz-Signature=calculated-signature
```

Minta contoh tanda tangan

Contoh permintaan pen AWS andatanganan berikut menunjukkan cara menggunakan SIGv4 untuk menandatangani permintaan yang dikirim tanpa AWS SDK atau alat AWS baris perintah.

Unggahan Amazon S3 berbasis browser menggunakan HTTP POST

Permintaan Otentik Browser-Based asi: Unggahan menjelaskan tanda tangan dan informasi relevan yang digunakan Amazon S3 untuk menghitung tanda tangan setelah menerima permintaan.

Contoh: Browser-Based Mengunggah menggunakan HTTP POST (Menggunakan AWS Tanda Tangan Versi 4) memberikan informasi selengkapnya dengan contoh kebijakan POST dan formulir yang dapat Anda gunakan untuk mengunggah file. Contoh kebijakan dan kredenSIAL fiktif menunjukkan alur kerja dan tanda tangan dan hash kebijakan yang dihasilkan.

Permintaan terautentikasi VPC Lattice

Contoh untuk permintaan terautentikasi Signature Version 4 (SIGv4) menyediakan contoh Python dan Java yang menunjukkan bagaimana Anda dapat melakukan penandatanganan permintaan dengan dan tanpa pencegat khusus.

Menggunakan Tanda Tangan Versi 4 dengan Amazon Translate

Terjemahan Langsung di Metaverse menunjukkan cara membangun aplikasi yang menghasilkan solusi terjemahan hampir real-time. Solusi penerjemah ucapan-ke-ucapan ini menggunakan AWS SIGv4 dalam pengkodean aliran acara untuk menghasilkan transkripsi waktu nyata.

Menggunakan Tanda Tangan Versi 4 dengan Neptunus

Contoh: Menghubungkan ke Neptunus Menggunakan Python dengan Tanda Tangan Versi 4 Penandatanganan menunjukkan cara membuat permintaan yang ditandatangani ke Neptunus menggunakan Python. Contoh ini mencakup variasi untuk menggunakan kunci akses atau kredenSIAL sementara.

Menandatangani permintaan HTTP ke Amazon Glacier

Contoh Perhitungan Tanda Tangan untuk Streaming API memandu Anda melalui detail pembuatan tanda tangan untuk Upload Archive (POST archive), salah satu dari dua API streaming di Amazon Glacier.

Membuat Permintaan HTTP ke Amazon SWF

Membuat Permintaan HTTP ke Amazon SWF menampilkan konten header untuk permintaan JSON ke Amazon SWF.

Perhitungan tanda tangan untuk streaming API di Amazon OpenSearch Service

Menandatangani permintaan pencarian OpenSearch Layanan Amazon dengan AWS SDK untuk PHP Versi 3 mencakup contoh cara mengirim permintaan HTTP yang ditandatangani ke Amazon OpenSearch Service.

Contoh proyek di AWS repositori sampel

Contoh proyek berikut menunjukkan cara menandatangani permintaan untuk membuat permintaan Rest API ke AWS layanan dengan bahasa umum seperti Python,, Java Node.js, C #, Go dan Rust.

Proyek Tanda Tangan Versi 4a

Proyek sigv4-signing-examples memberikan contoh cara menandatangani permintaan dengan SIGv4 untuk membuat permintaan Rest API dengan bahasa umum seperti Python,, Java, C # Node.js, Go dan Rust. Layanan AWS

Proyek sigv4a-signing-examples menyediakan contoh untuk menandatangani permintaan API Multi-wilayah, misalnya Access Points di Amazon S3. Multi-Region

Publikasikan ke AWS IoT Core

Kode Python untuk dipublikasikan AWS IoT Core menggunakan protokol HTTP s memberikan panduan tentang cara mempublikasikan pesan AWS IoT Core menggunakan protokol HTTPS dan otentikasi AWS SIGv4. Ini memiliki dua implementasi referensi - satu di Python dan lainnya di NodeJs.

Aplikasi.Net Framework untuk dipublikasikan AWS IoT Core menggunakan protokol HTTPs memberikan panduan tentang cara mempublikasikan pesan AWS IoT Core menggunakan protokol HTTPS dan otentikasi AWS SIGv4. Proyek ini juga mencakup implementasi ekivalen.NET core.

Memecahkan masalah penandatanganan Tanda Tangan Versi 4 untuk AWS Permintaan API

> **Important**
>
> Kecuali Anda menggunakan AWS SDK atau CLI, Anda harus menulis kode untuk menghitung tanda tangan yang memberikan informasi otentikasi dalam permintaan Anda. Perhitungan tanda tangan SIGv4 dapat menjadi tugas yang kompleks, dan kami sarankan Anda menggunakan AWS SDK atau CLI bila memungkinkan.

Saat Anda mengembangkan kode yang membuat permintaan yang ditandatangani, Anda mungkin menerima HTTP 403 `SignatureDoesNotMatch` dari Layanan AWS. Kesalahan ini berarti bahwa nilai tanda tangan dalam permintaan HTTP Anda AWS tidak cocok dengan tanda tangan yang di Layanan AWS hitung. Kes `Unauthorized` alahan HTTP 401 kembali ketika izin tidak mengizinkan pemanggil untuk membuat permintaan.

Permintaan API mungkin mengembalikan kesalahan jika:

- Permintaan API tidak ditandatangani dan permintaan API menggunakan otentikasi IAM.
- KredenSIAL IAM yang digunakan untuk menandatangani permintaan salah atau tidak memiliki izin untuk memanggil API.
- Tanda tangan permintaan API yang ditandatangani tidak cocok dengan tanda tangan yang dihitung AWS layanan.
- Header permintaan API salah.

> **Note**
>
> Perbarui protokol penandatanganan Anda dari Sig AWS nature versi 2 (SIGv2) ke Sig AWS nature versi 4 (SIGv4) sebelum menjelajahi solusi kesalahan lainnya. Layanan, seperti Amazon S3, dan Wilayah tidak lagi mendukung penandatanganan SIGv2.

Kemungkinan penyebab

- Kesalahan kredensi
- Permintaan kanonik dan kesalahan string penandatanganan

- Kesalahan ruang lingkup kredensi
- Kesalahan penandatanganan kunci

Kesalahan kredensi

Pastikan permintaan API ditandatangani dengan SIGv4. Jika permintaan API tidak ditandatangani, maka Anda mungkin menerima kesalahan:`Missing Authentication Token`. Tambahkan tanda tangan yang hilang dan kirim ulang permintaan.

Pastikan kredentif otentikasi untuk kunci akses dan kunci rahasia sudah benar. Jika kunci akses salah, maka Anda mungkin menerima kesalahan:`Unauthorized`. Pastikan entitas yang digunakan untuk menandatangani permintaan diberi wewenang untuk membuat permintaan. Lihat perinciannya di Memecahkan masalah pesan kesalahan akses ditolak.

Permintaan kanonik dan kesalahan string penandatanganan

Jika Anda salah menghitung permintaan kanonik di Buat hash permintaan kanonik atauBuat string untuk ditandatangani, langkah verifikasi tanda tangan yang dilakukan oleh layanan gagal dengan pesan kesalahan:

```
The request signature we calculated does not match the signature you provided
```

Ketika AWS layanan menerima permintaan yang ditandatangani, ia menghitung ulang tanda tangan. Jika ada perbedaan dalam nilai, maka tanda tangan tidak cocok. Bandingkan permintaan kanonik dan string dengan permintaan yang ditandatangani dengan nilai dalam pesan kesalahan. Ubah proses penandatanganan jika ada perbedaan.

> **Note**
>
> Anda juga dapat memverifikasi bahwa Anda tidak mengirim permintaan melalui proxy yang mengubah header atau permintaan.

Example Contoh permintaan kanonik

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
```

```
                                                    -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                       -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com      -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                        -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501        -------- Hash
 of the payload
```

Untuk memverifikasi bahwa kunci rahasia cocok dengan ID kunci akses, Anda dapat mengujinya dengan implementasi kerja yang diketahui. Misalnya, gunakan AWS SDK atau AWS CLI untuk membuat permintaan ke AWS.

Header permintaan API

Ketika header otorisasi kosong, kunci kredensi atau tanda tangan hilang atau salah, header tidak dimulai dengan nama algoritma, atau pasangan nilai kunci tidak menyertakan tanda sama, Anda menerima salah satu kesalahan berikut:

- Header otorisasi tidak boleh kosong.
- Header otorisasi memerlukan parameter 'Credential'.
- Header otorisasi membutuhkan parameter 'Tanda tangan'.
- Tanda tangan berisi pasangan key=value yang tidak valid (tanda sama hilang) di header Otorisasi.

Pastikan bahwa header otorisasi SIGv4 yang Anda tambahkan menyer Hitung tanda tangan takan kunci kredensia yang benar, dan juga menyertakan tanggal permintaan menggunakan Tanggal HTTP atau header. `x-amz-date`

Jika Anda menerima IncompleteSignatureException kesalahan dan konstruksi tanda tangan benar, Anda dapat memverifikasi bahwa header otorisasi tidak dimodifikasi saat transit ke Layanan AWS dengan menghitung SHA-256 hash dan pengkodean B64 dari header otorisasi dalam permintaan sisi klien Anda.

1. Dapatkan header otorisasi yang Anda kirim dalam permintaan. Header otorisasi Anda tampak mirip dengan contoh berikut:

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

2. Hitung SHA-256 hash dari header otorisasi.

```
hashSHA256(rawAuthorizationHeader) = hashedAuthorizationHeader
```

3. Enkode header otorisasi hash ke format Base64.

```
base64(hashedAuthorizationHeader) = encodedHashedAuthorizationHeader
```

4. Bandingkan string hash dan dikodekan yang baru saja Anda hitung dengan string yang Anda terima dalam pesan kesalahan Anda. Pesan kesalahan Anda harus mirip dengan contoh berikut:

```
com.amazon.coral.service#IncompleteSignatureException:
The signature contains an in-valid key=value pair (missing equal-sign)
in Authorization header (hashed with SHA-256 and encoded with Base64):
'9c574f83b4b950926da4a99c2b43418b3db8d97d571b5e18dd0e4f3c3ed1ed2c'.
```

- Jika dua hash berbeda, maka beberapa bagian dari header otorisasi berubah saat transit. Perubahan ini bisa disebabkan oleh penangan jaringan atau klien Anda yang melampirkan header yang ditandatangani atau mengubah header otorisasi dengan cara tertentu.
- Jika dua hash cocok, header otorisasi yang Anda kirim dalam permintaan cocok dengan apa yang AWS diterima. Tinjau pesan kesalahan yang Anda terima untuk menentukan apakah masalah tersebut disebabkan oleh kredentif atau tanda tangan yang salah. Kesalahan ini tercakup di bagian lain di halaman ini.

Kesalahan ruang lingkup kredensi

Cakupan kredensi yang Anda buat Buat string untuk ditandatangani membatasi tanda tangan ke tanggal, Wilayah, dan layanan tertentu. String ini memiliki format berikut:

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Jika Anda menggunakan Sigv4a, Wilayah tidak termasuk dalam cakupan kredensi.

Date

Jika cakupan kredensi tidak menentukan tanggal yang sama dengan x-amz-date header, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Jika permintaan menentukan waktu di masa mendatang, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Signature not yet current: date is still later than date
```

Jika permintaan telah kedaluwarsa, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Signature expired: date is now earlier than date
```

Region

Jika cakupan kredensi tidak menentukan Wilayah yang sama dengan permintaan, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Credential should be scoped to a valid Region, not region-code
```

Layanan

Jika cakupan kredensi tidak menentukan layanan yang sama dengan host header, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Credential should be scoped to correct service: 'service'
```

String penghentian

Jika cakupan kredensi tidak berakhir denganaws4_request, langkah verifikasi tanda tangan gagal dengan pesan kesalahan berikut:

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Kesalahan penandatanganan kunci

Kesalahan yang disebabkan oleh derivasi kunci penandatanganan yang salah atau penggunaan kriptografi yang tidak tepat lebih sulit untuk dipecahkan. Setelah memverifikasi bahwa string kanonik dan string yang akan ditandatangani sudah benar, Anda juga dapat memeriksa salah satu masalah berikut:

- Kunci akses rahasia tidak cocok dengan ID kunci akses yang Anda tentukan.
- Ada masalah dengan kode derivasi kunci Anda.

Untuk memverifikasi bahwa kunci rahasia cocok dengan ID kunci akses, Anda dapat mengujinya dengan implementasi kerja yang diketahui. Misalnya, gunakan AWS SDK atau AWS CLI untuk membuat permintaan ke AWS. Sebagai contoh, lihat Minta contoh tanda tangan

Referensi kebijakan IAM JSON

Bagian ini menyajikan sintaks, deskripsi, dan contoh elemen, variabel, dan logika evaluasi kebijakan JSON yang terperinci di IAM. Untuk informasi umum selengkapnya, lihat Gambaran dari kebijakan JSON.

Referensi ini mencakup bagian berikut.

- Referensi elemen kebijakan IAM JSON — Pelajari lebih banyak tentang elemen yang dapat Anda gunakan ketika membuat kebijakan. Lihat contoh kebijakan tambahan dan pelajari tentang ketentuan, jenis data yang didukung, dan bagaimana digunakan dalam berbagai layanan.
- Logika evaluasi kebijakanBagian ini menjelaskan AWS permintaan, bagaimana mereka diautentikasi, dan cara AWS menggunakan kebijakan untuk menentukan akses ke sumber daya.
- Tata bahasa kebijakan IAM JSON — Bagian ini menyajikan tata bahasa formal untuk bahasa yang digunakan untuk membuat kebijakan di IAM.
- AWS kebijakan terkelola untuk fungsi pekerjaan— Bagian ini mencantumkan semua kebijakan AWS terkelola yang secara langsung dipetakan ke fungsi pekerjaan umum di industri TI. Gunakan kebijakan ini untuk memberikan izin yang diperlukan untuk melaksanakan tugas yang diharapkan

dari seseorang dalam fungsi pekerjaan tertentu. Kebijakan ini mengonsolidasikan izin untuk banyak layanan ke dalam satu kebijakan.

- AWS kunci konteks kondisi globalBagian ini mencakup daftar semua kunci kondisi AWS global yang dapat Anda gunakan untuk membatasi izin dalam kebijakan IAM.
- IAM dan AWS STS kunci konteks kondisiBagian ini mencakup daftar semua kunci IAM dan AWS STS kondisi yang dapat Anda gunakan untuk membatasi izin dalam kebijakan IAM.
- Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan — Bagian ini menyajikan daftar semua operasi AWS API yang dapat Anda gunakan sebagai izin dalam kebijakan IAM. Ini juga mencakup kunci kondisi khusus layanan yang dapat digunakan untuk lebih menyempurnakan permintaan.

Referensi elemen kebijakan IAM JSON

Dokumen kebijakan JSON terdiri atas elemen-elemen. Elemen-elemen ini tercantum di sini dalam urutan umum Anda menggunakannya dalam kebijakan. Urutan elemen tidak penting—misalnya, elemen `Resource` dapat muncul sebelum elemen `Action`. Anda tidak diwajibkan untuk menentukan elemen `Condition` apa pun dalam kebijakan. Untuk mempelajari selengkapnya tentang struktur dan tujuan umum dokumen kebijakan JSON, lihat Gambaran dari kebijakan JSON.

Beberapa elemen kebijakan JSON saling eksklusif. Ini berarti bahwa Anda tidak dapat membuat kebijakan yang menggunakan keduanya. Misalnya, Anda tidak dapat menggunakan `Action` dan `NotAction` dalam pernyataan kebijakan yang sama. Pasangan lain yang saling eksklusif termasuk `Principal`/`NotPrincipal` dan `Resource`/`NotResource`.

Detail tentang apa saja yang termasuk ke dalam kebijakan berbeda-beda untuk setiap layanan, tergantung pada tindakan apa yang disediakan layanan, tipe sumber daya apa yang ada di dalamnya, dan sebagainya. Ketika Anda menulis kebijakan untuk layanan tertentu, akan sangat membantu jika Anda melihat contoh kebijakan untuk layanan tersebut. Untuk daftar semua layanan yang mendukung IAM, dan untuk tautan ke dokumentasi dalam layanan tersebut yang membahas IAM dan kebijakan, lihat AWS layanan yang bekerja dengan IAM.

Saat Anda membuat atau mengedit kebijakan JSON, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan Anda lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk

mempelajari selengkapnya tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Topik

- Elemen kebijakan IAM JSON: Version
- Elemen kebijakan IAM JSON: Id
- Elemen kebijakan IAM JSON: Statement
- Elemen kebijakan IAM JSON: Sid
- Elemen kebijakan IAM JSON: Efek
- AWS Elemen kebijakan JSON: Principal
- AWS Elemen kebijakan JSON: NotPrincipal
- Elemen kebijakan IAM JSON: Tindakan
- Elemen kebijakan IAM JSON: NotAction
- Elemen kebijakan IAM JSON: Sumber daya
- Elemen kebijakan IAM JSON: NotResource
- Elemen kebijakan IAM JSON: Kondisi
- Elemen kebijakan IAM: Variabel dan tag
- Elemen kebijakan JSON IAM: Jenis data yang didukung

Elemen kebijakan IAM JSON: Version

> **Catatan disambiguasi**
>
> Elemen kebijakan JSON `Version` ini berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, di sisi lain, dibuat ketika Anda membuat perubahan pada kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola. Jika Anda mencari informasi tentang dukungan beberapa versi yang tersedia untuk kebijakan terkelola, lihat the section called “Peningkatan versi IAM”.

Elemen kebijakan `Version` menetapkan aturan sintaksis bahasa yang akan digunakan untuk memproses kebijakan. Untuk menggunakan semua fitur kebijakan yang tersedia, sertakan `Version` elemen berikut di luar `Statement` elemen di semua kebijakan Anda.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM mendukung nilai elemen `Version` berikut:

- `2012-10-17`. Ini adalah versi bahasa kebijakan saat ini, dan Anda harus selalu menyertakan elemen `Version` dan mengaturnya ke `2012-10-17`. Jika tidak, Anda tidak dapat menggunakan fitur seperti variabel kebijakan yang diperkenalkan dengan versi ini.
- `2008-10-17`. Ini adalah versi terdahulu dari bahasa kebijakan. Anda mungkin melihat versi ini pada kebijakan lama yang sudah ada. Jangan gunakan versi ini untuk kebijakan baru atau saat Anda memperbarui kebijakan yang sudah ada. Fitur yang lebih baru, seperti variabel kebijakan, tidak akan berfungsi dengan kebijakan Anda. Misalnya, variabel seperti `${aws:username}` tidak diakui sebagai variabel dan diperlakukan sebagai string literal di dalam kebijakan.

Elemen kebijakan IAM JSON: Id

Elemen `Id` ini menentukan pengidentifikasi opsional untuk kebijakan. ID digunakan secara berbeda di layanan yang berbeda. ID diperbolehkan dalam kebijakan berbasis sumber daya, tetapi tidak dalam kebijakan berbasis identitas.

Untuk layanan yang memungkinkan Anda mengatur elemen `ID` kami menyarankan Anda menggunakan UUID (GUID) untuk nilai, atau menyertakan UUID sebagai bagian dari ID untuk memastikan keunikan.

JSON

```
{
  "Version":"2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Beberapa AWS layanan (misalnya, Amazon SQS atau Amazon SNS) mungkin memerlukan elemen ini dan memiliki persyaratan keunikan untuk itu. Untuk informasi khusus layanan tentang penulisan kebijakan, lihat dokumentasi untuk layanan yang sedang Anda kerjakan.

Elemen kebijakan IAM JSON: Statement

Elemen `Statement` adalah elemen utama kebijakan ini. Elemen ini wajib diisi. Elemen `Statement` dapat berisi satu pernyataan atau serangkaian pernyataan individu. Setiap blok pernyataan individu harus ditutup dengan tanda kurung kurawal { }. Untuk beberapa pernyataan, larik harus dilampirkan dalam kurung persegi [ ].

```
"Statement": [{...},{...},{...}]
```

Contoh berikut menunjukkan kebijakan yang berisi susunan tiga pernyataan di dalam satu elemen `Statement` tunggal. (Kebijakan ini memungkinkan Anda mengakses “folder rumah” Anda sendiri di konsol Amazon S3.) Kebijakan ini mencakup variabel `aws:username`, yang diganti selama evaluasi kebijakan dengan nama pengguna dari permintaan. Lihat informasi yang lebih lengkap di Pengantar.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Elemen kebijakan IAM JSON: Sid

Anda dapat memberikan `Sid` (ID pernyataan) sebagai pengenal opsional untuk pernyataan kebijakan. Anda dapat menetapkan nilai `Sid` untuk setiap pernyataan dalam rangkaian pernyataan. Anda dapat menggunakan `Sid` nilai sebagai deskripsi untuk pernyataan kebijakan. Dalam layanan yang memungkinkan Anda menentukan `ID` elemen, seperti SQS dan SNS, `Sid` nilainya hanyalah sub-ID dari ID dokumen kebijakan. Di dalam IAM, nilai `Sid` harus unik dalam kebijakan JSON.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

`Sid`Elemen ini mendukung huruf besar ASCII (A-Z), huruf kecil (a-z), dan angka (0-9).

IAM tidak mengekspos API `Sid` IAM. Anda tidak dapat mengambil pernyataan tertentu berdasarkan ID ini.

> **Note**
>
> Beberapa AWS layanan (misalnya, Amazon SQS atau Amazon SNS) mungkin memerlukan elemen ini dan memiliki persyaratan keunikan untuk itu. Untuk informasi khusus layanan tentang kebijakan penulisan, lihat dokumentasi untuk layanan tempat Anda bekerja.

Elemen kebijakan IAM JSON: Efek

Elemen `Effect` diperlukan dan menentukan apakah pernyataan tersebut mengakibatkan diizinkan atau ditolak. Nilai yang valid untuk `Effect` adalah `Allow` dan `Deny`. Nilainya `Effect` peka huruf besar/kecil.

```
"Effect":"Allow"
```

Secara default, akses ke sumber daya ditolak. Untuk mengizinkan akses ke sumber daya, Anda harus mengatur elemen `Effect` ke `Allow`. Untuk menimpa izinkan (misalnya, untuk menimpa izinkan yang berlaku sebaliknya), Anda mengatur elemen `Effect` ke `Deny`. Lihat informasi yang lebih lengkap di Logika evaluasi kebijakan.

AWS Elemen kebijakan JSON: Principal

Gunakan `Principal` elemen dalam kebijakan JSON berbasis sumber daya untuk menentukan prinsip yang diizinkan atau ditolak akses ke sumber daya.

Anda harus menggunakan `Principal` elemen dalam kebijakan berbasis sumber daya. Beberapa layanan mendukung kebijakan berbasis sumber daya, termasuk IAM. Jenis kebijakan berbasis sumber daya IAM adalah kebijakan kepercayaan peran. Dalam peran IAM, gunakan `Principal` elemen dalam kebijakan kepercayaan peran untuk menentukan siapa yang dapat mengambil peran tersebut. Untuk akses akun silang, Anda harus menentukan pengidentifikasi 12-digit dari akun tepercaya. Untuk mempelajari apakah prinsipal dalam akun di luar zona kepercayaan (organisasi atau akun terpercaya) memiliki akses untuk mengasumsikan peran Anda, lihat Apa yang dimaksud dengan Penganalisis Akses IAM?.

> **Note**
>
> Setelah membuat peran, Anda dapat mengubah akun menjadi "*" agar semua orang dapat mengambil peran tersebut. Jika Anda melakukannya, kami sangat menyarankan Anda untuk membatasi siapa yang dapat mengakses peran tersebut melalui cara lain, seperti elemen `Condition` yang membatasi akses ke alamat IP tertentu saja. Jangan biarkan peran Anda dapat diakses semua orang!

Contoh sumber daya lain yang mendukung kebijakan berbasis sumber daya termasuk bucket Amazon S3 atau. AWS KMS key

Anda tidak dapat menggunakan `Principal` elemen dalam kebijakan berbasis identitas. Identity-based kebijakan adalah kebijakan izin yang Anda lampirkan ke identitas IAM (pengguna, grup, atau peran). Dalam kasus tersebut, prinsipal secara implisit adalah identitas di mana kebijakan dilampirkan.

Topik

- Cara menentukan prinsipal
- Akun AWS utama
- Prinsip peran IAM
- Prinsip sesi peran
- Prinsipal federasi OIDC

- Prinsipal federasi SAML
- Prinsip pengguna IAM
- Kepala Pusat Identitas IAM
- AWS STS prinsipal pengguna federasi
- AWS prinsipal layanan
- AWS prinsipal layanan di Wilayah opt-in
- Semua kepala sekolah
- Informasi selengkapnya

Cara menentukan prinsipal

Anda menentukan prinsipal dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

Anda dapat menyebutkan salah satu prinsip dasar berikut dalam kebijakan:

- Akun AWS dan pengguna root
- Peran IAM
- Sesi peran
- Pengguna IAM:
- Prinsip pengguna federasi
- AWS layanan
- Semua kepala sekolah

Anda tidak dapat mengidentifikasi grup pengguna sebagai prinsipal dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup berhubungan dengan izin, bukan otentikasi, dan prinsipal adalah entitas IAM yang diautentikasi.

Anda dapat menentukan lebih dari satu prinsipal untuk masing-masing tipe prinsipal dalam bagian berikut menggunakan array. Susunan dapat mengambil satu atau beberapa nilai. Bila Anda menentukan lebih dari satu prinsipal dalam elemen, Anda memberikan izin untuk setiap prinsipal. Ini logis `OR` dan tidak logis`AND`, karena Anda mengotentikasi sebagai satu kepala sekolah pada satu waktu. Jika Anda menyertakan lebih dari satu nilai, gunakan tanda kurung persegi (`[`dan`]`) dan batasi koma setiap entri untuk array. Kebijakan contoh berikut menentukan izin untuk akun 123456789012 atau akun 555555555555.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Anda tidak dapat menggunakan wildcard untuk mencocokkan sebagian nama pengguna utama atau ARN.

Akun AWS utama

Anda dapat menentukan Akun AWS pengidentifikasi dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ini mendelegasikan otoritas ke akun. Ketika Anda mengizinkan akses ke akun yang berbeda, administrator di akun tersebut kemudian harus memberikan akses ke identitas (pengguna atau peran IAM) di akun tersebut. Ketika Anda menentukan Akun AWS, Anda dapat menggunakan akun ARN (arn:aws:iam: ::root*account-ID*), atau bentuk singkat yang terdiri dari awalan diikuti oleh ID akun. `"AWS":`

Misalnya, memberikan ID akun `123456789012`, Anda dapat menggunakan dari metode berikut untuk menyebutkan akun tersebut di elemen `Principal`:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

ARN akun dan ID akun yang dipersingkat berperilaku dengan cara yang sama. Keduanya mendelegasikan izin ke akun. Menggunakan akun ARN dalam `Principal` elemen tidak membatasi izin hanya untuk pengguna root akun.

> **Note**
>
> Saat Anda menyimpan kebijakan berbasis sumber daya yang menyertakan ID akun yang dipersingkat, layanan mungkin mengubahnya menjadi ARN utama. Ini tidak mengubah fungsionalitas kebijakan.

Beberapa AWS layanan mendukung opsi tambahan untuk menentukan prinsipal akun. Misalnya, Amazon S3 memungkinkan Anda menentukan ID pengguna canonik menggunakan format berikut:

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Anda juga dapat menentukan lebih dari satu Akun AWS, (atau ID pengguna kanonik) sebagai prinsipal menggunakan array. Misalnya, Anda dapat menentukan prinsipal dalam kebijakan bucket menggunakan ketiga metode tersebut.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Prinsip peran IAM

Anda dapat menentukan ARN utama peran IAM dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Peran IAM adalah identitas. Di IAM, identitas adalah sumber daya yang dapat Anda tetapkan izin. Peran mempercayai identitas lain yang diautentikasi untuk mengambil peran itu. Ini termasuk prinsipal AWS atau pengguna dari penyedia identitas eksternal (IdP). Ketika kepala sekolah atau identitas mengambil peran, mereka menerima kredentif keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredentif sesi tersebut untuk melakukan operasi AWS, mereka menjadi kepala sesi peran.

Saat Anda menentukan prinsipal peran dalam kebijakan berbasis sumber daya, izin efektif untuk prinsipal dibatasi oleh jenis kebijakan apa pun yang membatasi izin untuk peran tersebut. Ini termasuk kebijakan sesi dan batas izin. Untuk informasi selengkapnya tentang bagaimana izin efektif untuk sesi peran dievaluasi, lihatLogika evaluasi kebijakan.

Untuk menentukan peran ARN dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Jika `Principal` elemen Anda dalam kebijakan kepercayaan peran berisi ARN yang menunjuk ke peran IAM tertentu, maka ARN tersebut berubah menjadi ID utama peran yang unik saat Anda menyimpan kebijakan. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat ulang peran. Anda biasanya tidak melihat ID ini di konsol, karena IAM menggunakan transformasi terbalik kembali ke peran ARN saat kebijakan kepercayaan ditampilkan.
> Namun, jika Anda menghapus peran, maka Anda memutuskan hubungan. Kebijakan tidak lagi berlaku, bahkan jika Anda membuat ulang peran tersebut karena peran baru memiliki ID utama baru yang tidak cocok dengan ID yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul dalam kebijakan berbasis sumber daya karena tidak AWS dapat lagi memetakannya kembali ke ARN yang valid.
> Hasilnya adalah jika Anda menghapus dan membuat ulang peran yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran dalam kebijakan untuk mengganti ID utama dengan ARN yang benar. ARN sekali lagi berubah menjadi ID utama peran yang baru saat Anda menyimpan kebijakan. Untuk informasi selengkapnya, lihat Mem ahami AWS Penanganan peran IAM yang Dihapus di Kebijakan.

Atau, Anda dapat menentukan prinsip peran sebagai prinsipal dalam kebijakan berbasis sumber daya atau membuat kebijakan izin luas yang menggunakan kunci kondisi. `aws:PrincipalArn` Bila Anda menggunakan kunci ini, prinsip sesi peran diberikan izin berdasarkan ARN peran yang diasumsikan, dan bukan ARN sesi yang dihasilkan. Karena AWS tidak mengubah ARN kunci kondisi menjadi ID, izin yang diberikan ke peran ARN tetap ada jika Anda menghapus peran dan kemudian membuat peran baru dengan nama yang sama. Identity-based jenis kebijakan, seperti batas izin atau kebijakan sesi, tidak membatasi izin yang diberikan menggunakan kunci `aws:PrincipalArn` kondisi dengan wildcard (*) di `Principal` elemen, kecuali kebijakan berbasis identitas berisi penolakan eksplisit.

Prinsip sesi peran

Anda dapat menentukan sesi peran dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Ketika kepala sekolah atau identitas mengambil peran, mereka menerima kredentif keamanan sementara dengan izin peran yang diasumsikan. Ketika mereka menggunakan kredentif sesi tersebut untuk melakukan operasi AWS, mereka menjadi kepala sesi peran.

Format yang Anda gunakan untuk prinsip sesi peran tergantung pada AWS STS operasi yang digunakan untuk mengambil peran.

> **Important**
>
> AWS merekomendasikan penggunaan prinsi pal peran IAM dalam kebijakan Anda alih-alih prinsipal sesi peran jika memungkinkan. Gunakan `Condition` pernyataan dan kunci kondisi untuk memperluas cakupan akses bila diperlukan.

Untuk menentukan ARN utama sesi peran dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Selain itu, administrator dapat merancang proses untuk mengontrol bagaimana sesi peran dikeluarkan. Misalnya, mereka dapat memberikan solusi satu klik untuk pengguna mereka yang membuat nama sesi yang dapat diprediksi. Jika administrator Anda melakukan ini, Anda dapat menggunakan prinsipal sesi peran dalam kebijakan atau kunci kondisi Anda. Jika tidak, Anda dapat menentukan peran ARN sebagai prinsipal dalam kunci `aws:PrincipalArn` kondisi. Cara Anda menentukan peran sebagai kepala sekolah dapat mengubah izin efektif untuk sesi yang dihasilkan. Untuk informasi selengkapnya, lihat Prinsip peran IAM.

Prinsipal federasi OIDC

Prinsip federasi OIDC adalah prinsip yang digunakan saat memanggil AWS STS `AssumeRoleWithWebIdentity` API dengan token web JSON (JWT) dari IDP yang sesuai dengan OIDC, juga dikenal sebagai OpenID Provider (OP), untuk meminta kredenSIAL sementara. AWS Prinsipal federasi OIDC dapat mewakili IDP OIDC di AWS akun Anda, atau 4 penyedia identitas bawaan:,,, Login with Amazon GoogleFacebook, dan Amazon Cognito. https://docs.aws.amazon.com/cognito/latest/developerguide/role-based-access-control.html

Pengguna, beban kerja, atau sistem yang telah mengeluarkan JWT dari IDP OIDC mereka dapat memanggil `AssumeRoleWithWebIdentity` menggunakan JWT untuk meminta kredenSIAL AWS keamanan sementara untuk peran IAM yang dikonfigurasi untuk mempercayai IDP OIDC yang mengeluarkan JWT. JWT dapat berupa token id, token akses, atau token JWT yang dikirimkan dengan metode lain selama memenuhi persyaratan yang tercantum oleh. AWS STS Untuk informasi selengkapnya, lihat S kenario umum dan Meminta kredenSIAL melalui penyedia OIDC.

Gunakan tipe utama ini dalam kebijakan kepercayaan peran Anda untuk mengizinkan atau menolak izin panggilan `AssumeRoleWIthWebIdentity` menggunakan IDP OIDC yang ada di Anda Akun AWS, atau salah satu dari empat IDP bawaan. Untuk menentukan ARN utama federasi OIDC dalam `Principal` elemen kebijakan kepercayaan peran, gunakan salah satu dari empat format berikut untuk IDP OIDC bawaan:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Saat menggunakan penyedia OIDC yang Anda tambahkan ke akun Anda, misalnya GitHub, Anda menentukan ARN penyedia dalam kebijakan kepercayaan peran Anda. Konfigurasi ini memungkinkan Anda menulis kebijakan IAM yang mengontrol akses khusus untuk pengguna yang diautentikasi melalui penyedia identitas kustom Anda.

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/full-OIDC-
identity-provider-URL" }
```

Misalnya, jika GitHub penyedia identitas web tepercaya, ARN sesi peran OIDC dalam `Principal` elemen kebijakan kepercayaan peran menggunakan format berikut:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/
tokens.actions.githubusercontent.com" }
```

Lihat Meng onfigurasi OpenID Connect di Amazon Web Services untuk informasi selengkapnya.

Prinsipal federasi OIDC tidak didukung dalam jenis kebijakan selain kebijakan kepercayaan peran.

Prinsipal federasi SAML

Prinsip federasi SAML adalah prinsip yang digunakan saat memanggil AWS STS `AssumeRoleWithSAML` API untuk meminta AWS kredenSIAL sementara menggunakan pernyataan

SAML. Anda dapat menggunakan penyedia identitas SAML (IDP) untuk masuk, dan kemudian mengambil peran IAM menggunakan operasi ini. Mirip dengan`AssumeRoleWithWebIdentity`, `AssumeRoleWithSAML` tidak memerlukan AWS kredenSIAL untuk otentikasi. Sebagai gantinya, pengguna pertama-tama mengotentikasi dengan penyedia identitas SAML mereka, kemudian melakukan panggilan `AssumeRoleWithSAML` API menggunakan pernyataan SAML mereka, atau diarahkan ke AWS Sign-In/SAML halaman untuk masuk ke. Konsol Manajemen AWS Untuk informasi selengkapnya tentang prinsipal mana yang dapat mengambil peran menggunakan operasi ini, lihatBandingkan AWS STS kredensialnya.

Gunakan tipe utama ini dalam kebijakan kepercayaan peran Anda untuk mengizinkan atau menolak izin berdasarkan penyedia identitas SAML tepercaya. Untuk menentukan sesi peran identitas SAML ARN dalam `Principal` elemen kebijakan kepercayaan peran, gunakan format berikut:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

Prinsip pengguna IAM

Anda dapat menentukan pengguna IAM dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

> **Note**
>
> Dalam `Principal` elemen, bagian nama pengguna dari Amazon Resource Name (ARN) peka huruf besar/kecil.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Saat Anda menentukan pengguna di elemen `Principal`, Anda tidak dapat menggunakan wildcard (*) yang berarti “semua pengguna”. Kepala sekolah harus selalu memberi nama pengguna tertentu.

> **Important**
>
> Jika `Principal` elemen Anda dalam kebijakan kepercayaan peran berisi ARN yang menunjuk ke pengguna IAM tertentu, maka IAM mengubah ARN menjadi ID utama unik pengguna saat Anda menyimpan kebijakan. Hal ini membantu memitigasi risiko seseorang meningkatkan hak istimewa mereka dengan menghapus dan membuat ulang pengguna. Anda biasanya tidak melihat ID ini di konsol, karena juga ada transformasi balik kembali ke ARN pengguna ketika kebijakan kepercayaan ditampilkan.
> Namun, jika Anda menghapus pengguna, maka Anda memutuskan hubungan. Kebijakan tidak lagi berlaku, bahkan saat Anda membuat ulang pengguna. Itu karena pengguna baru memiliki ID utama baru yang tidak cocok dengan ID yang disimpan dalam kebijakan kepercayaan. Ketika ini terjadi, ID utama muncul dalam kebijakan berbasis sumber daya karena tidak AWS dapat lagi memetakannya kembali ke ARN yang valid.
> Hasilnya adalah jika Anda menghapus dan membuat ulang pengguna yang direferensikan dalam `Principal` elemen kebijakan kepercayaan, Anda harus mengedit peran untuk mengganti ID utama yang sekarang salah dengan ARN yang benar. IAM sekali lagi mengubah ARN menjadi ID utama pengguna yang baru saat Anda menyimpan kebijakan.

Kepala Pusat Identitas IAM

Di Pusat Identitas IAM, prinsipal dalam kebijakan berbasis sumber daya harus didefinisikan sebagai prinsipal. Akun AWS Untuk menentukan akses, referensi peran ARN dari izin yang ditetapkan dalam blok kondisi. Untuk detailnya, lihat Referensi kumpulan izin di kebijakan sumber daya, Amazon EKS, dan AWS KMS di Panduan Pengguna Pusat Identitas IAM.

AWS STS prinsipal pengguna federasi

Anda dapat menentukan sesi pengguna federasi dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal.

> **Important**
>
> AWS menyarankan agar Anda membatasi penggunaan sesi pengguna AWS STS federasi. Sebagai gantinya, gunakan peran https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_cross-account-with-roles.html IAM.

Prinsip pengguna AWS STS federasi dibuat melalui `GetFederationToken` operasi yang dipanggil dengan kredenSIAL IAM yang berumur panjang. Izin pengguna federasi adalah persimpangan dari prinsipal yang dipanggil `GetFederationToken` dan kebijakan sesi yang diteruskan sebagai parameter ke `GetFederationToken` API.

Di AWS, pengguna IAM atau pengguna Pengguna root akun AWS dapat mengotentikasi menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya tentang prinsipal mana yang dapat berfederasi menggunakan operasi ini, lihat. Bandingkan AWS STS kredensialnya

- Pengguna federasi IAM — Pengguna IAM melakukan federasi menggunakan `GetFederationToken` operasi yang menghasilkan sesi pengguna federasi untuk pengguna IAM tersebut.
- Pengguna root federasi — Pengguna root berfederasi menggunakan `GetFederationToken` operasi yang menghasilkan sesi pengguna federasi untuk pengguna root tersebut.

Ketika pengguna IAM atau pengguna root meminta kredentif sementara dari AWS STS menggunakan operasi ini, mereka memulai sesi pengguna federasi sementara. ARN sesi ini didasarkan pada identitas asli yang disatukan.

Untuk menentukan sesi pengguna federasi ARN dalam `Principal` elemen, gunakan format berikut:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS prinsipal layanan

Anda dapat menentukan AWS layanan dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Prinsi p layanan adalah pengidentifikasi untuk layanan.

Peran IAM yang dapat diasumsikan oleh AWS layanan disebut peran ??? layanan. Peran layanan harus menyertakan kebijakan kepercayaan. Kebijakan kepercayaan adalah kebijakan berbasis sumber daya yang melekat pada peran yang menentukan kepala sekolah mana yang dapat mengambil peran tersebut. Beberapa peran layanan telah menetapkan kebijakan kepercayaan. Namun, dalam beberapa kasus, Anda harus menentukan prinsip utama layanan dalam kebijakan kepercayaan. Prinsip layanan dalam kebijakan IAM tidak bisa`"Service": "*"`.

> **Important**
>
> Pengidentifikasi untuk prinsipal layanan mencakup nama layanan, dan biasanya dalam format berikut:
> *service-name*.amazonaws.com

Prinsipal layanan ditentukan oleh layanan. Anda dapat menemukan prinsipal layanan untuk beberapa layanan dengan membukaAWS layanan yang bekerja dengan IAM, memeriksa apakah layanan memiliki Ya di kolom Service-linked peran, dan membuka tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut. Temukan bagian Service-Linked Izin Peran untuk layanan tersebut untuk melihat prinsipal layanan.

Contoh berikut menunjukkan kebijakan yang dapat dilampirkan pada peran layanan. Kebijakan tersebut memungkinkan dua layanan, Amazon ECS dan Elastic Load Balancing, untuk mengambil peran tersebut. Layanan kemudian dapat melakukan tugas yang diberikan oleh kebijakan izin yang ditetapkan untuk peran tersebut (tidak ditampilkan). Untuk menetapkan beberapa prinsipal layanan, Anda tidak menentukan dua elemen `Service`; Anda hanya dapat memiliki satu. Sebagai gantinya, Anda menggunakan serangkaian dari beberapa prinsipal layanan sebagai nilai elemen `Service` tunggal.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS prinsipal layanan di Wilayah opt-in

Anda dapat meluncurkan sumber daya di beberapa Wil AWS ayah dan beberapa Wilayah yang harus Anda ikuti. Untuk daftar lengkap Wilayah yang harus Anda ikuti, lihat M engelola AWS Wilayah di Referensi Umum AWS panduan.

Ketika AWS layanan di Wilayah opt-in membuat permintaan dalam Wilayah yang sama, format nama utama layanan diidentifikasi sebagai versi non-regional dari nama utama layanan mereka:

service-name.amazonaws.com

Ketika AWS layanan di Wilayah opt-in membuat permintaan lintas wilayah ke Wilayah lain, format nama utama layanan diidentifikasi sebagai versi regional dari nama utama layanan mereka:

service-name.*{region}*.amazonaws.com

Misalnya, Anda memiliki topik Amazon SNS yang terletak di Wilayah ap-southeast-1 dan bucket Amazon S3 yang terletak di Wilayah opt-in. ap-east-1 Anda ingin mengonfigurasi notifikasi bucket S3 untuk menerbitkan pesan ke topik SNS. Untuk mengizinkan layanan S3 memposting pesan ke topik SNS, Anda harus memberikan sns:Publish izin utama layanan S3 melalui kebijakan akses berbasis sumber daya dari topik tersebut.

Jika Anda menentukan versi non-regional dari prinsip layanan S3,s3.amazonaws.com, dalam kebijakan akses topik, sns:Publish permintaan dari bucket ke topik akan gagal. Contoh berikut menentukan prinsip layanan S3 non-regional dalam elemen Principal kebijakan kebijakan kebijakan akses topik SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Karena bucket terletak di Wilayah opt-in dan permintaan dibuat di luar Wilayah yang sama, prinsipal layanan S3 muncul sebagai nama utama layanan regional,. s3.ap-east-1.amazonaws.com Anda harus menggunakan nama utama layanan regional saat layanan di Wil AWS ayah opt-in mengajukan permintaan ke Wilayah lain. Setelah Anda menentukan nama utama layanan regional, jika bucket membuat sns:Publish permintaan ke topik SNS yang terletak di Wilayah lain, permintaan akan berhasil. Contoh berikut menentukan prinsip layanan S3 regional dalam elemen Principal kebijakan kebijakan kebijakan akses topik SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Kebijakan sumber daya atau daftar izin berbasis prinsipal layanan untuk permintaan Lintas wilayah dari Wilayah pilihan ke Wilayah lain hanya akan berhasil jika Anda menentukan nama utama layanan regional.

> **Note**
>
> Untuk kebijakan kepercayaan peran IAM, sebaiknya gunakan nama utama layanan yang tidak terregional. Sumber daya IAM bersifat global dan oleh karena itu peran yang sama dapat digunakan di Wilayah mana pun.

Semua kepala sekolah

Anda dapat menggunakan wildcard (*) untuk menentukan semua prinsipal dalam `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Resource-based kebijakanizin pemberian dan kunci kondisi digunakan untuk membatasi kondisi pernyataan kebijakan.

> **Important**
>
> Kami sangat menyarankan agar Anda tidak menggunakan wildcard (*) dalam `Principal` elemen kebijakan berbasis sumber daya dengan `Allow` efek kecuali Anda bermaksud memberikan akses publik atau anonim. Jika tidak, tentukan prinsipal, layanan, atau AWS akun yang dimaksud dalam `Principal` elemen dan kemudian membatasi akses lebih lanjut dalam `Condition` elemen. Hal ini terutama berlaku untuk kebijakan kepercayaan peran IAM, karena kebijakan tersebut memungkinkan prinsipal lain untuk menjadi prinsipal di akun Anda.

Untuk kebijakan berbasis sumber daya, menggunakan wildcard (*) dengan `Allow` efek memberikan akses ke semua pengguna, termasuk pengguna anonim (akses publik). Untuk pengguna IAM dan kepala peran dalam akun Anda, tidak diperlukan izin lain. Untuk kepala sekolah di akun lain, mereka juga harus memiliki izin berbasis identitas di akun mereka yang memungkinkan mereka mengakses sumber daya Anda. Ini disebut akses https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic-cross-account.html lintas akun.

Untuk pengguna anonim, elemen-elemen berikut setara:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Anda tidak dapat menggunakan wildcard untuk mencocokkan sebagian nama pengguna utama atau ARN.

Contoh berikut menunjukkan kebijakan berbasis sumber daya yang dapat digunakan alih-alih secara eksplisit AWS Elemen kebijakan JSON: NotPrincipal menolak semua prinsipal kecuali yang ditentukan dalam elemen. `Condition` Kebijakan ini harus ditambahkan ke bucket Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Informasi selengkapnya

Untuk informasi selengkapnya, lihat berikut ini:

- Contoh kebijakan bucket di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- Contoh kebijakan untuk Amazon SNS di Panduan Pengembang Layanan Pemberitahuan Sederhana Amazon
- Contoh kebijakan Amazon SQS di Panduan Pengembang Layanan Antrian Sederhana Amazon
- Kebijakan utama dalam Panduan Peng AWS Key Management Service embang
- Pengidentifikasi akun di Referensi Umum AWS
- Federasi OIDC

AWS Elemen kebijakan JSON: NotPrincipal

`NotPrincipal`Elemen menggunakan `"Effect":"Deny"` untuk menolak akses ke semua prinsipal kecuali prinsipal yang ditentukan dalam elemen. `NotPrincipal` Prinsipal dapat berupa pengguna IAM, prinsipal pengguna AWS STS federasi, peran IAM, sesi peran yang diasumsikan,, AWS layanan Akun AWS, atau jenis utama lainnya. Untuk informasi lebih lanjut tentang kepala sekolah, lihat. AWS Elemen kebijakan JSON: Principal

`NotPrincipal`harus digunakan dengan`"Effect":"Deny"`. Menggunakannya `"Effect":"Allow"` dengan tidak didukung.

> **Important**
>
> Kami tidak merekomendasikan penggunaan `NotPrincipal` untuk kebijakan berbasis sumber daya baru sebagai bagian dari strategi keamanan dan otorisasi Anda. Saat Anda menggunakan`NotPrincipal`, pemecahan masalah efek dari beberapa jenis kebijakan bisa jadi sulit. Sebaiknya gunakan kunci `aws:PrincipalArn` konteks dengan operator kondisi ARN sebagai gantinya.

Poin kunci

- `NotPrincipal`Elemen ini didukung dalam kebijakan berbasis sumber daya untuk beberapa layanan AWS , termasuk titik akhir VPC. Kebijakan berbasis sumber daya adalah kebijakan yang diterapkan langsung ke sumber daya. Anda tidak dapat menggunakan `NotPrincipal` elemen dalam kebijakan berbasis identitas IAM atau dalam kebijakan kepercayaan peran IAM.
- Jangan gunakan pernyataan kebijakan berbasis sumber daya yang menyertakan elemen `NotPrincipal` kebijakan dengan `Deny` efek untuk pengguna IAM atau peran yang memiliki kebijakan batas izin yang dilampirkan. `NotPrincipal`Elemen dengan `Deny` efek akan selalu menolak prinsip IAM yang memiliki kebijakan batas izin yang dilampirkan, terlepas dari nilai yang ditentukan dalam elemen. `NotPrincipal` Hal ini menyebabkan beberapa pengguna IAM atau peran yang seharusnya memiliki akses ke sumber daya kehilangan akses. Sebaiknya ubah pernyataan kebijakan berbasis sumber daya Anda untuk menggunakan operator kondisi `ArnNotEquals`dengan kunci `aws:PrincipalArn`konteks untuk membatasi akses, bukan elemen. `NotPrincipal` Untuk informasi tentang batas izin, lihatBatas izin untuk entitas IAM.
- Saat Anda menggunakan`NotPrincipal`, Anda juga harus menentukan ARN akun dari prinsipal yang tidak ditolak. Jika tidak, kebijakan ini dapat menolak akses ke seluruh akun yang berisi prinsipal. Bergantung pada layanan yang Anda sertakan dalam kebijakan Anda, AWS dapat

memvalidasi akun terlebih dahulu, lalu pengguna. Jika pengguna peran yang diasumsikan (seseorang yang menggunakan peran) sedang dievaluasi, AWS mungkin memvalidasi akun terlebih dahulu, kemudian peran, dan kemudian pengguna peran yang dianggap. Pengguna peran yang diasumsikan diidentifikasi oleh nama sesi peran yang ditentukan saat mereka mengasumsikan peran. Oleh karena itu, kami sangat menyarankan Anda secara eksplisit untuk menyertakan ARN untuk akun pengguna, atau menyertakan ARN untuk peran dan ARN untuk akun yang berisi peran tersebut.

- `NotPrincipal`Elemen tidak didukung dalam Kebijakan Kontrol Layanan (SCP) dan Kebijakan Kontrol Sumber Daya (RCP).

Alternatif untuk **`NotPrincipal`** elemen

Saat mengelola kontrol akses di AWS, mungkin ada skenario di mana Anda perlu secara eksplisit menolak semua akses prinsipal ke sumber daya, kecuali untuk satu atau beberapa prinsip yang Anda tentukan. AWS merekomendasikan penggunaan pernyataan Deny dengan kunci konteks kondisi global untuk kontrol yang lebih tepat dan pemecahan masalah yang lebih mudah. Contoh berikut menunjukkan pendekatan alternatif menggunakan operator kondisi seperti `StringNotEquals` atau `ArnNotEquals` untuk menolak akses ke semua prinsipal kecuali yang ditentukan dalam elemen Kondisi.

Contoh skenario menggunakan peran IAM

Anda dapat menggunakan kebijakan berbasis sumber daya dengan pernyataan Deny untuk mencegah semua peran IAM, kecuali yang ditentukan dalam elemen Kondisi, mengakses atau memanipulasi sumber daya Anda. Pendekatan ini mengikuti prinsip AWS keamanan bahwa penolakan eksplisit selalu diutamakan daripada pernyataan izin apa pun dan membantu mempertahankan prinsip hak istimewa terkecil di seluruh infrastruktur Anda. AWS

Alih-alih menggunakan`NotPrincipal`, sebaiknya gunakan pernyataan Deny dengan kunci konteks kondisi global dan operator kondisi `ArnNotEquals`ingin secara eksplisit mengizinkan akses peran IAM ke sumber daya Anda. Contoh berikut digunakan aws: PrincipalArn untuk secara eksplisit mengizinkan peran `read-only-role` mengakses bucket Amazon S3 di folder. `Bucket_Account_Audit`

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "DenyCrossAuditAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::Bucket_Account_Audit",
        "arn:aws:s3:::Bucket_Account_Audit/*"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:role/read-only-role"
        }
      }
    }
  ]
}
```

Contoh skenario menggunakan prinsip layanan

Anda dapat menggunakan pernyataan Deny untuk mencegah semua prinsip layanan, kecuali yang ditentukan dalam `Condition` elemen, mengakses atau memanipulasi sumber daya Anda. Pendekatan ini sangat berguna ketika Anda perlu menerapkan kontrol akses berbutir halus atau menetapkan batas keamanan antara berbagai layanan dan aplikasi di lingkungan Anda. AWS

Alih-alih menggunakan`NotPrincipal`, sebaiknya gunakan pernyataan Deny dengan kunci konteks kondisi global dan operator kondisi `StringNotEquals`untuk secara eksplisit mengizinkan akses utama layanan ke sumber daya Anda. Contoh berikut digunakan `aws: PrincipalServiceName` untuk secara eksplisit mengizinkan prinsipal AWS CodeBuild layanan mengakses bucket Amazon S3 di folder. `BUCKETNAME`

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNotCodeBuildAccess",
      "Effect": "Deny",
```

```
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME",
        "arn:aws:s3:::BUCKETNAME/*"
      ],
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:PrincipalServiceName": "codebuild.amazonaws.com"
        }
      }
    }
  ]
}
```

Elemen kebijakan IAM JSON: Tindakan

Elemen `Action` menguraikan tindakan khusus atau tindakan yang akan diizinkan atau ditolak. Pernyataan harus mencakup elemen `Action` atau `NotAction`. Setiap AWS layanan memiliki serangkaian tindakan sendiri yang menggambarkan tugas yang dapat Anda lakukan dengan layanan itu. Misalnya, daftar tindakan untuk Amazon S3 dapat ditemukan di Menentukan Iz in dalam Kebijakan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon, daftar tindakan untuk Amazon EC2 dapat ditemukan di Referensi API Amazon EC2, dan daftar tindakan untuk AWS Identity and Access Management dapat ditemukan di Referensi API IAM. Untuk menemukan daftar tindakan untuk layanan lain, lihat dokumentasi referensi API untuk layanan.

AWS juga menyediakan informasi referensi layanan dalam format JSON untuk merampingkan otomatisasi alur kerja manajemen kebijakan. Dengan informasi referensi layanan, Anda dapat mengakses tindakan, sumber daya, dan kunci kondisi yang tersedia di seluruh file yang Layanan AWS dapat dibaca mesin. Untuk informasi selengkapnya, lihat Layanan AWS Informasi yang disederhanakan untuk akses terprogram di Referensi Otorisasi Layanan.

Anda menentukan nilai menggunakan namespace layanan sebagai prefiks tindakan (`iam`, `ec2`, `sqs`, `sns`, `s3`, dll.) diikuti dengan nama tindakan yang mengizinkan atau menolak. Nama harus sesuai dengan tindakan yang didukung oleh layanan. Prefiks dan nama tindakan tidak sensitif dengan huruf besar-kecil. Misalnya, `iam:ListAccessKeys` sama dengan `IAM:listaccesskeys`. Contoh berikut menunjukkan elemen `Action` untuk layanan yang berbeda.

Tindakan Amazon SQS

```
"Action": "sqs:SendMessage"
```

Tindakan Amazon EC2

```
"Action": "ec2:StartInstances"
```

Tindakan IAM

```
"Action": "iam:ChangePassword"
```

Tindakan Amazon S3

```
"Action": "s3:GetObject"
```

Anda dapat menentukan beberapa nilai untuk elemen `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Anda dapat menggunakan wildcard pencocokan multi-karakter (*) dan wildcard pencocokan karakter tunggal (?) untuk memberikan akses ke semua tindakan yang ditawarkan produk tertentu. AWS Misalnya, elemen `Action` berikut berlaku untuk semua tindakan S3.

```
"Action": "s3:*"
```

Anda juga dapat menggunakan wildcard (*atau?) sebagai bagian dari nama tindakan. Misalnya, elemen `Action` berikut berlaku untuk semua tindakan IAM yang mencakup string `AccessKey`, termasuk `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys`, dan `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Beberapa layanan memungkinkan Anda membatasi tindakan yang tersedia. Misalnya, Amazon SQS memungkinkan Anda menyediakan hanya sebagian kecil dari semua kemungkinan tindakan Amazon SQS. Dalam hal ini, wildcard * tidak memungkinkan kontrol penuh atas antrean; itu hanya memungkinkan subset tindakan yang telah Anda bagikan. Untuk informasi lebih lanjut, lihat Memahami Izin di Panduan Pengembang Amazon Simple Queue Service.

Elemen kebijakan IAM JSON: NotAction

`NotAction` adalah elemen kebijakan tingkat lanjut yang secara eksplisit cocok dengan semuanya kecuali daftar tindakan yang ditentukan. Menggunakan `NotAction` dapat menghasilkan kebijakan yang lebih singkat dengan mencantumkan hanya beberapa tindakan yang harus tidak cocok, daripada menyertakan daftar panjang tindakan yang sesuai. Tindakan yang `NotAction` ditentukan dalam tidak terpengaruh oleh `Allow` atau `Deny` efek dalam pernyataan kebijakan. Hal ini, pada gilirannya, berarti bahwa semua tindakan atau layanan yang berlaku yang tidak terdaftar diperbolehkan jika Anda menggunakan efek `Allow`. Sebagai tambahan, tindakan atau layanan yang tidak terdaftar tersebut ditolak jika Anda menggunakan efek `Deny`. Saat Anda menggunakan `NotAction` dengan elemen `Resource`, Anda memberikan cakupan untuk kebijakan. Ini adalah bagaimana AWS menentukan tindakan atau layanan mana yang berlaku. Untuk informasi selengkapnya, lihat kebijakan contoh berikut.

NotAction dengan Izinkan

Anda dapat menggunakan `NotAction` elemen dalam pernyataan dengan `"Effect": "Allow"` untuk menyediakan akses ke semua tindakan dalam AWS layanan, kecuali untuk tindakan yang ditentukan dalam`NotAction`. Anda dapat menggunakannya dengan elemen `Resource` untuk menyediakan cakupan untuk kebijakan, membatasi tindakan yang diizinkan untuk tindakan yang dapat dilakukan pada sumber daya tertentu.

Contoh berikut memungkinkan pengguna mengakses semua tindakan Amazon S3 yang dapat dilakukan pada sumber daya S3 kecuali untuk menghapus bucket. Kebijakan ini juga tidak mengizinkan tindakan dalam layanan lain, karena tindakan layanan lainnya tidak berlaku untuk sumber daya S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Terkadang, Anda mungkin ingin memungkinkan akses ke sejumlah besar tindakan. Dengan menggunakan elemen `NotAction` secara efektif membalik pernyataan, yang menghasilkan daftar tindakan yang lebih singkat. Misalnya, karena AWS memiliki begitu banyak layanan, Anda mungkin ingin membuat kebijakan yang memungkinkan pengguna melakukan semuanya kecuali mengakses tindakan IAM.

Contoh berikut memungkinkan pengguna untuk mengakses setiap tindakan di setiap AWS layanan kecuali IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Hati-hati saat menggunakan elemen `NotAction` dan `"Effect": "Allow"` dalam pernyataan yang sama atau dalam pernyataan yang berbeda dalam kebijakan. `NotAction` cocok dengan semua layanan dan tindakan yang tidak tercantum atau berlaku secara eksplist untuk sumber daya yang ditentukan, dan dapat mengakibatkan pemberian izin kepada pengguna lebih dari yang Anda inginkan.

NotAction dengan mendustakan

Anda dapat menggunakan elemen `NotAction` dalam pernyataan dengan `"Effect": "Deny"` untuk menolak akses ke semua sumber daya yang tercantum kecuali untuk tindakan yang ditentukan dalam elemen `NotAction`. Kombinasi ini tidak memungkinkan item yang terdaftar, tetapi sebaliknya secara eksplisit menolak tindakan yang tidak tercantum. Anda tetap harus mengizinkan tindakan yang ingin Anda izinkan.

Contoh kondisi berikut menolak akses ke tindakan non-IAM jika pengguna tidak masuk menggunakan MFA. Jika pengguna masuk dengan MFA, maka `"Condition"` pengujian gagal dan pernyataan akhir `"Deny"` tidak berpengaruh. Namun demikian, harap diperhatikan bahwa hal ini tidak akan memberikan akses kepada pengguna ke tindakan apa pun; hal ini hanya akan secara eksplisit menolak semua tindakan lainnya kecuali tindakan IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Untuk contoh kebijakan yang menolak akses ke tindakan di luar Wilayah tertentu, kecuali untuk tindakan dari layanan tertentu, lihat AWS: Menolak akses AWS berdasarkan Wilayah yang diminta.

Elemen kebijakan IAM JSON: Sumber daya

`Resource`Elemen dalam pernyataan kebijakan IAM mendefinisikan objek atau objek yang diterapkan pernyataan tersebut. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`.

Anda menentukan sumber daya menggunakan Nama Sumber Daya Amazon (ARN). Format ARN tergantung pada Layanan AWS dan sumber daya spesifik yang Anda maksud. Meskipun format ARN bervariasi, Anda selalu menggunakan ARN untuk mengidentifikasi sumber daya. Untuk informasi selengkapnya tentang format ARN, lihat ARN IAM. Untuk informasi tentang cara menentukan sumber daya, lihat dokumentasi untuk layanan yang ingin Anda tulis pernyataan.

> **Note**
>
> Beberapa Layanan AWS tidak mengizinkan Anda menentukan tindakan untuk sumber daya individu. Dalam kasus ini, tindakan apa pun yang Anda daftarkan di `NotAction` elemen `Action` atau berlaku untuk semua sumber daya dalam layanan tersebut. Ketika ini terjadi, Anda menggunakan karakter wildcard (*) dalam `Resource` elemen.

Contoh berikut mengacu pada antrean Amazon SQS tertentu.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

Contoh berikut mengacu pada pengguna IAM yang Akun AWS dinamai Bob dalam.

> **Note**
>
> Dalam `Resource` elemen, nama pengguna IAM peka huruf besar/kecil.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Menggunakan wildcard di ARN sumber daya

Anda dapat menggunakan karakter wildcard (*dan?) dalam segmen individu ARN (bagian yang dipisahkan oleh titik dua) untuk mewakili:

- Kombinasi karakter apa pun (*)

- Setiap karakter tunggal (?)

Anda dapat menggunakan beberapa * atau ? karakter di setiap segmen. Jika * wildcard adalah karakter terakhir dari segmen ARN sumber daya, karakter tersebut dapat diperluas untuk mencocokkan di luar batas titik dua. Sebaiknya gunakan wildcard (*dan?) dalam segmen ARN yang dipisahkan oleh titik dua.

> **Note**
>
> Anda tidak dapat menggunakan wildcard di segmen layanan yang mengidentifikasi AWS produk. Untuk informasi selengkapnya tentang segmen ARN, lihat Identifikasi AWS sumber daya dengan Nama Sumber Daya Amazon (ARN)

Contoh berikut mengacu pada semua pengguna IAM yang alurnya `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

Contoh berikut mengacu pada semua item dalam bucket Amazon S3 spesifik.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
```

Karakter asterisk (*) dapat diperluas untuk menggantikan semua yang ada di dalam segmen, termasuk karakter seperti garis miring maju (/) yang mungkin tampak sebagai pembatas dalam namespace layanan tertentu. Misalnya, pertimbangkan Amazon S3 ARN berikut sebagai logika ekspansi wildcard yang sama berlaku untuk semua layanan.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*/test/*"
```

Wildcard di ARN berlaku untuk semua objek berikut dalam bucket, tidak hanya objek pertama yang terdaftar.

```
amzn-s3-demo-bucket/1/test/object.jpg
amzn-s3-demo-bucket/1/2/test/object.jpg
amzn-s3-demo-bucket/1/2/test/3/object.jpg
amzn-s3-demo-bucket/1/2/3/test/4/object.jpg
amzn-s3-demo-bucket/1///test///object.jpg
amzn-s3-demo-bucket/1/test/.jpg
amzn-s3-demo-bucket//test/object.jpg
```

```
amzn-s3-demo-bucket/1/test/
```

Pertimbangkan dua objek terakhir dalam daftar sebelumnya. Nama objek Amazon S3 dapat dimulai atau diakhiri dengan karakter pembatas konvensional forward slash (). Meskipun / berfungsi sebagai pembatas, tidak ada signifikansi khusus ketika karakter ini digunakan dalam sumber daya ARN. Hal ini diperlakukan sama dengan karakter valid lainnya. ARN tidak akan cocok dengan objek berikut:

```
amzn-s3-demo-bucket/1-test/object.jpg
amzn-s3-demo-bucket/test/object.jpg
amzn-s3-demo-bucket/1/2/test.jpg
```

Menentukan beberapa sumber daya

Anda dapat menentukan beberapa sumber daya dalam `Resource` elemen dengan menggunakan array ARN. Contoh berikut mengacu pada dua tabel DynamoDB.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Menggunakan variabel kebijakan dalam ARN sumber daya

Di elemen `Resource`, Anda dapat menggunakan variabel kebijakan JSON pada bagian ARN yang mengidentifikasi sumber daya spesifik (yaitu, dalam bagian jejak ARN). Misalnya, Anda dapat menggunakan tombol `{aws:username}` sebagai bagian dari ARN sumber daya untuk menunjukkan bahwa nama pengguna saat ini harus disertakan sebagai bagian dari nama sumber daya. Contoh berikut menunjukkan bagaimana Anda dapat menggunakan tombol `{aws:username}` di elemen `Resource`. Kebijakan ini memungkinkan akses ke tabel Amazon DynamoDB yang cocok dengan nama pengguna saat ini.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "dynamodb:*",
```

```
        "Resource": "arn:aws:dynamodb:us-east-2:111122223333:table/
${aws:username}"
    }
}
```

Untuk informasi selengkapnya tentang variabel kebijakan JSON, lihat Elemen kebijakan IAM: Variabel dan tag.

Elemen kebijakan IAM JSON: NotResource

`NotResource` adalah elemen kebijakan tingkat lanjut yang secara eksplisit cocok dengan setiap sumber daya kecuali yang ditentukan. Menggunakan `NotResource` dapat menghasilkan kebijakan yang lebih pendek dengan mencantumkan hanya beberapa sumber yang seharusnya tidak sesuai, daripada menyertakan daftar panjang sumber daya yang sesuai. Ini khususnya berguna untuk kebijakan yang menerapkan satu layanan AWS .

Misalnya, bayangkan Anda memiliki grup bernama `HRPayroll`. Anggota tidak `HRPayroll` boleh diizinkan mengakses sumber daya Amazon S3 apa pun kecuali `Payroll` folder di `HRBucket` bucket. Kebijakan berikut secara jelas menolak akses ke semua sumber daya Amazon S3 selain sumber daya yang tercantum. Namun demikian, harap diperhatikan bahwa kebijakan ini tidak memberikan akses ke sumber daya apa pun kepada pengguna.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

Biasanya, untuk secara eksplisit menolak akses ke sumber daya yang akan Anda tuliskan kebijakan yang menggunakan `"Effect":"Deny"` dan yang menyertakan elemen `Resource` yang

mencantumkan setiap folder secara terpisah. Namun, dalam hal ini, setiap kali Anda menambahkan folder keHRBucket, atau menambahkan sumber daya ke Amazon S3 yang tidak boleh diakses, Anda harus menambahkan namanya ke daftar di Resource elemen. Jika Anda menggunakan elemen NotResource sebagai gantinya, akses pengguna secara otomatis ditolak ke folder baru kecuali Anda menambahkan nama folder untuk elemen NotResource.

Saat menggunakan NotResource, Anda harus ingat bahwa sumber daya yang ditentukan dalam elemen ini hanya tindakan yang terbatas. Hal ini, pada gilirannya, membatasi semua sumber daya yang akan berlaku untuk tindakan tersebut. Dalam contoh di atas, kebijakan hanya memengaruhi tindakan Amazon S3 dan karena itu hanya sumber daya Amazon S3. Jika Action elemen juga menyertakan tindakan Amazon EC2, kebijakan akan menolak akses ke sumber daya EC2 apa pun yang tidak ditentukan dalam elemen. NotResource Untuk mempelajari tindakan mana dalam layanan yang memungkinkan menentukan ARN sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

NotResource dengan elemen lain

Jangan pernah menggunakan elemen "Effect": "Allow", "Action": "*", dan "NotResource": "arn:aws:s3:::HRBucket" secara bersama-sama. Pernyataan ini sangat berbahaya, karena memungkinkan semua tindakan masuk AWS pada semua sumber daya kecuali bucket HRBucket S3. Ini bahkan memungkinkan pengguna menambahkan kebijakan yang memungkinkan mereka mengakses HRBucket. Jangan lakukan ini.

Hati-hati saat menggunakan elemen NotResource dan "Effect": "Allow" dalam pernyataan yang sama atau dalam pernyataan yang berbeda dalam kebijakan. NotResource memungkinkan semua layanan dan sumber daya yang tidak tercantum secara eksplist, dan dapat mengakibatkan pemberian izin kepada pengguna lebih dari yang Anda inginkan. Menggunakan elemen NotResource dan "Effect": "Deny" dalam pernyataan yang sama menolak layanan dan sumber daya yang secara tidak eksplisit tercantum.

Elemen kebijakan IAM JSON: Kondisi

Elemen Condition (atau blok Condition) memungkinkan Anda menentukan ketentuan saat kebijiakan berlaku. Elemen Condition bersifat opsional. Di Condition elemen, Anda membangun ekspresi di mana Anda menggunakan operator kondisi (sama, kurang dari, dan lainnya) untuk mencocokkan kunci konteks dan nilai dalam kebijakan dengan kunci dan nilai dalam konteks permintaan. Untuk mempelajari selengkapnya tentang konteks permintaan, lihat Komponen permintaan.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

Kunci konteks yang Anda tentukan dalam kondisi kebijakan dapat berupa kunci konteks kondisi global atau kunci konteks khusus layanan. Kunci konteks kondisi global memiliki `aws:` awalan. Service-specific kunci konteks memiliki awalan layanan. Misalnya, Amazon EC2 memungkinkan Anda menulis kondisi menggunakan kunci `ec2:InstanceType` konteks, yang unik untuk layanan tersebut. Untuk melihat kunci konteks IAM khusus layanan dengan `iam:` awalan, lihat. IAM dan AWS STS kunci konteks kondisi

Nama kunci konteks tidak p eka huruf besar/kecil. Misalnya, memasukkan kunci `aws:SourceIP` konteks setara dengan pengujian untuk`AWS:SourceIp`. Case-sensitivity nilai kunci konteks tergantung pada operator kondisi yang Anda gunakan. Misalnya, kondisi berikut menyertakan `StringEquals` operator untuk memastikan bahwa hanya permintaan yang dibuat oleh kecoco `john` kan. Nama pengguna `John` ditolak aksesnya.

```
"Condition" : { "StringEquals" : { "aws:username" : "john" }}
```

Ketentuan berikut menggunakan operator `StringEqualsIgnoreCase` untuk mencocokkan nama pengguna `john` atau `John`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "john" }}
```

Beberapa kunci konteks mendukung pasangan kunci-nilai yang memungkinkan Anda menentukan bagian dari nama kunci. Contohnya termasuk kunci `aws:RequestTag/tag-key` konteks, kunci konteks AWS KMS `kms:EncryptionContext:encryption_context_key`, dan kunci `ResourceTag/tag-key` konteks yang didukung oleh beberapa layanan.

- Jika Anda menggunakan kunci `ResourceTag/tag-key` konteks untuk layanan seperti Amazon EC2, maka Anda harus menentukan nama kunci untuk. `tag-key`
- Nama kunci tidak peka huruf besar/kecil. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya.
- AWS layanan yang mendukung atribut ini mungkin memungkinkan Anda membuat beberapa nama kunci yang berbeda hanya berdasarkan kasus. Misalnya, Anda dapat menandai instans Amazon EC2 dengan `ec2=test1` dan`EC2=test2`. Saat Anda menggunakan kondisi seperti `"aws:ResourceTag/EC2": "test1"` untuk memungkinkan akses ke sumber daya tersebut,

nama kunci cocok dengan kedua tanda, tetapi hanya satu nilai yang cocok. Hal ini dapat mengakibatkan kegagalan ketentuan yang tidak terduga.

> **Important**
>
> Sebagai praktik terbaik, pastikan bahwa anggota akun Anda mengikuti konvensi pemberian nama yang konsisten ketika memberi nama atribut pasangan nilai-kunci. Contohnya mencakup tanda atau konteks enkripsi AWS KMS . Anda dapat menerapkan ini menggunakan kunci `aws:TagKeys` konteks untuk penandaan, atau `kms:EncryptionContextKeys` untuk konteks en AWS KMS kripsi.

- Untuk daftar semua operator kondisi dan deskripsi cara kerjanya, lihat Oper ator kondisi.
- Kecuali ditentukan lain, semua kunci konteks dapat memiliki beberapa nilai. Untuk penjelasan tentang cara menangani kunci konteks yang memiliki banyak nilai, lihatMengatur operator untuk kunci konteks multinilai.
- Untuk daftar semua kunci konteks yang tersedia secara global, lihatAWS kunci konteks kondisi global.
- Untuk kunci konteks kondisi yang ditentukan oleh setiap layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan.

Konteks permintaan

Ketika kepala sekolah membuat permintaan ke AWS, AWS mengumpulkan informasi permintaan ke dalam konteks permintaan. Konteks permintaan mencakup informasi tentang prinsip, sumber daya, tindakan, dan properti lingkungan lainnya. Evaluasi kebijakan mencocokkan properti dalam kebijakan dengan properti yang dikirim dalam permintaan untuk mengevaluasi dan mengotorisasi tindakan yang dapat Anda lakukan. AWS

Anda dapat menggunakan `Condition` elemen kebijakan JSON untuk menguji kunci konteks tertentu terhadap konteks permintaan. Misalnya, Anda dapat membuat kebijakan yang menggunakan kunci CurrentTime konteks aws: untuk memungkinkan pengguna melakukan tindakan hanya dalam rentang tanggal tertentu.

Contoh berikut menunjukkan representasi konteks permintaan ketika Martha Rivera mengirim permintaan untuk menonaktifkan perangkat MFA-nya.

```
Principal: AROA123456789EXAMPLE
Action: iam:DeactivateMFADevice
Resource: arn:aws:iam::user/martha
Context:
  - aws:UserId=AROA123456789EXAMPLE:martha
  - aws:PrincipalAccount=1123456789012
  - aws:PrincipalOrgId=o-example
  - aws:PrincipalARN=arn:aws:iam::1123456789012:assumed-role/TestAR
  - aws:MultiFactorAuthPresent=true
  - aws:MultiFactorAuthAge=2800
  - aws:CurrentTime=...
  - aws:EpochTime=...
  - aws:SourceIp=...
```

Konteks permintaan dicocokkan dengan kebijakan yang memungkinkan pengguna menghapus perangkat otentikasi multi-faktor (MFA) mereka sendiri, tetapi hanya jika mereka telah masuk menggunakan MFA dalam satu jam terakhir (3.600 detik).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

Dalam contoh ini, kebijakan cocok dengan konteks permintaan: tindakannya sama, sumber daya cocok dengan wildcard “*”, dan nilainya `aws:MultiFactorAuthAge` adalah 2800, yang kurang dari 3600, sehingga kebijakan mengizinkan permintaan otorisasi ini.

AWS mengevaluasi setiap kunci konteks dalam kebijakan dan mengembalikan nilai true atau false. Kunci konteks yang tidak ada dalam permintaan dianggap sebagai ketidakcocokan.

Konteks permintaan dapat mengembalikan nilai-nilai berikut:

- Benar – Jika pemohon masuk menggunakan MFA dalam satu jam terakhir atau kurang, maka ketentuan kembali ke benar.
- Salah – Jika pemohon masuk menggunakan MFA lebih dari satu jam yang lalu, maka ketentuan kembali ke salah.
  - Tidak hadir — Jika pemohon membuat permintaan menggunakan kunci akses pengguna IAM mereka di AWS API AWS CLI atau, kunci tidak ada. Dalam hal ini, kuncinya tidak ada, dan kunci tidak akan cocok.

> **Note**
>
> Dalam beberapa kasus, ketika nilai kunci kondisi tidak ada, kondisi masih dapat mengembalikan true. Misalnya, jika Anda menambahkan ku `ForAllValues` alifikasi, permintaan mengembalikan true jika kunci konteks tidak ada dalam permintaan. Untuk mencegah kunci konteks yang hilang atau kunci konteks dengan nilai kosong dievaluasi menjadi true, Anda dapat menyertakan operator kondisi Null dalam kebijakan Anda dengan `false` nilai untuk memeriksa apakah kunci konteks ada dan nilainya tidak null.

Blok ketentuan

Contoh berikut menunjukkan format dasar dari elemen `Condition`:

```
"Condition": {"StringLike": {"s3:prefix": ["jane/*"]}}
```

Nilai dari permintaan diwakili oleh kunci konteks, dalam hal ini`s3:prefix`. Nilai kunci konteks dibandingkan dengan nilai yang Anda tentukan sebagai nilai literal, seperti`jane/*`. Jenis perbandingan yang akan dibuat ditentukan oleh operator kondisi (di sini,`StringLike`). Anda dapat membuat ketentuan yang membandingkan string, tanggal, angka, dan lainnya menggunakan perbandingan umum Boolean seperti sama, lebih besar dari, dan kurang dari. Bila Anda menggunakan operator string atau operator ARN, Anda juga dapat menggunakan variabel kebijakan dalam nilai kunci konteks. Contoh berikut termasuk `aws:username` variabel.

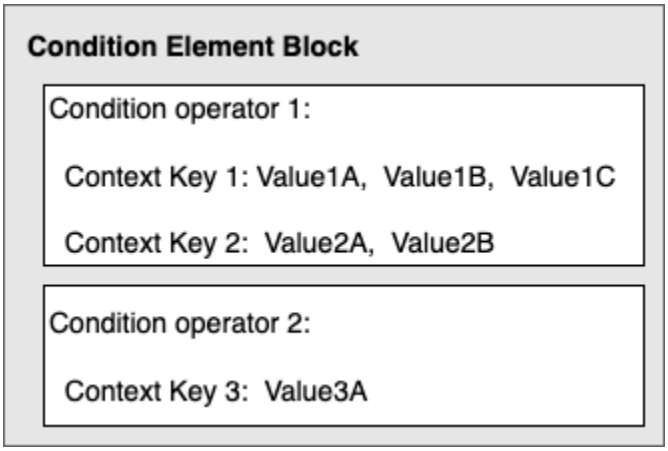
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Dalam beberapa keadaan, kunci konteks dapat berisi beberapa nilai. Misalnya, permintaan ke Amazon DynamoDB mungkin meminta untuk mengembalikan atau memperbarui beberapa atribut dari tabel. Kebijakan untuk akses ke tabel DynamoDB dapat menyertakan kunci `dynamodb:Attributes` konteks, yang berisi semua atribut yang tercantum dalam permintaan. Anda dapat menguji beberapa atribut dalam daftar permintaan terhadap daftar atribut yang diizinkan di kebijakan dengan menggunakan operator kumpulan di elemen `Condition`. Untuk informasi selengkapnya, lihat Mengatur operator untuk kunci konteks multinilai.

Ketika kebijakan dievaluasi selama permintaan, AWS ganti kunci dengan nilai yang sesuai dari permintaan. (Dalam contoh ini, AWS akan menggunakan tanggal dan waktu permintaan.) Kondisi dievaluasi untuk mengembalikan benar atau salah, yang kemudian diperhitungkan dalam apakah kebijakan tersebut secara keseluruhan mengizinkan atau menolak permintaan.

Beberapa nilai dalam suatu ketentuan

Sebuah `Condition` elemen dapat berisi beberapa operator kondisi, dan setiap operator kondisi dapat berisi beberapa pasangan kunci-nilai konteks. Gambar berikut mengilustrasikan hal ini.



Lihat informasi yang lebih lengkap di Mengatur operator untuk kunci konteks multinilai.

Elemen kebijakan JSON IAM: Operator kondisi

Gunakan operator ketentuan di elemen `Condition` untuk mencocokkan kunci kondisi dan nilai di dalam kebijakan terhadap nilai di dalam konteks permintaan. Untuk informasi selengkapnya tentang elemen `Condition`, lihat Elemen kebijakan IAM JSON: Kondisi.

Operator ketentuan yang dapat Anda gunakan dalam kebijakan tergantung pada kunci kondisi yang Anda pilih. Anda dapat memilih kunci kondisi global atau kunci kondisi khusus layanan. Untuk mempelajari operator ketentuan mana yang dapat Anda gunakan pada kunci kondisi global, lihat AWS kunci konteks kondisi global. Untuk mempelajari operator kondisi mana yang dapat Anda gunakan untuk kunci kondisi khusus layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan dan pilih layanan yang ingin Anda lihat.

> **Important**
>
> Jika kunci yang Anda tentukan dalam kondisi kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok dan kondisinya salah. Jika kondisi kebijakan mengharuskan kunci tidak cocok, seperti `StringNotLike` atau`ArnNotLike`, dan kunci yang tepat tidak ada, kondisinya benar. Logika ini berlaku untuk semua operator kondisi kecuali... IfExistsdan pemeriksaan Null. Operator ini menguji apakah kuncinya ada (exists) dalam konteks permintaan.

Operator ketentuan dapat dikelompokkan ke dalam kategori berikut:

- Tali
- Numerik
- Tanggal dan waktu
- Boolean
- biner
- Alamat IP
- Amazon Resource Name (ARN) (hanya tersedia untuk beberapa layanan.)
- ... IfExists(memeriksa apakah nilai kunci ada sebagai bagian dari pemeriksaan lain)
- Pemeriksaan null (memeriksa apakah nilai kunci ada sebagai cek mandiri)

Operator ketentuan string

Operator ketentuan string memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai string.

- Variabel kebijakan - Di dukung
- Wildcard - Didukung ???

Operator ketentuan	Deskripsi
`StringEquals`	Kecocokan yang tepat, peka terhadap huruf besar-kecil
`StringNotEquals`	Pencocokan dinegasikan
`StringEqualsIgnoreCase`	Pencocokan yang tepat, mengabaikan huruf besar-kecil
`StringNotEqualsIgnoreCase`	Pencocokan yang dinegasikan, mengabaikan huruf besar-kecil
`StringLike`	Case-sensitive mencocokkan. Nilai dapat mencakup wildcard pencocokan multi-karakter (*) dan wildcard pencocokan karakter tunggal (?) di mana saja di string. Anda harus menentukan wildcard untuk mencapai kecocokan string paruh. **Note** Jika kunci berisi beberapa nilai, `StringLike` dapat memenuhi syarat dengan operator set— `ForAllValues:StringLike` dan `ForAnyValue:StringLike` . Untuk informasi selengkapnya, lihat Mengatur operator untuk kunci konteks multinilai.
`StringNotLike`	Kecocokan dinegasikan kepekaan huruf besar-kecil. Nilai dapat mencakup wildcard pencocokan multi-karakter (*) atau wildcard pencocokan karakter tunggal (?) di mana saja di string.

Example operator kondisi string

Misalnya, pernyataan berikut berisi `Condition` elemen yang menggunakan `aws:PrincipalTag` kunci untuk menentukan bahwa prinsipal yang membuat permintaan harus diberi tag dengan kategori `iamuser-admin` pekerjaan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalTag/job-category": "iamuser-admin"
            }
        }
    }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Dalam contoh ini, kunci `aws:PrincipalTag/job-category` ada dalam konteks permintaan jika prinsipnya menggunakan pengguna IAM dengan tag terlampir. Juga disertakan untuk prinsipal menggunakan IAM role dengan tanda terlampir atau tanda sesi. Jika pengguna tanpa tanda mencoba melihat atau mengedit access key, ketentuan akan kembali `false` dan permintaan tersebut ditolak secara tersirat oleh pernyataan ini.

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - iamuser-admin`	Cocokkan
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin"`	`aws:PrincipalTag/job-category: - dev-ops`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
}		
"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }	Tidak aws:PrincipalTag/job-category dalam konteks permintaan.	Tidak cocok

Example menggunakan variabel kebijakan dengan operator kondisi string

Contoh berikut menggunakan operator ketentuan `StringLike` untuk melakukan pencocokan string dengan variabel kebijakan untuk membuat kebijakan yang memungkinkan pengguna IAM menggunakan konsol Amazon S3 untuk mengelola "direktori beranda"-nya sendiri dalam bucket Amazon S3. Kebijakan tersebut mengizinkan tindakan tertentu pada bucket S3 selama `s3:prefix` cocok dengan salah satu pola yang ditentukan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "",
            "home/",
```

```
            "home/${aws:username}/"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini untuk pengguna yang berbeda berdasarkan aws:username nilai dalam konteks permintaan.

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	`aws:username: - martha_rivera`	`"StringLike": { "s3:prefix": [ "home/", "home/martha_rivera /" ] }`
`"StringLike": { "s3:prefix": [ "home/", "home/${aws:username}/" ] }`	`aws:username: - nikki_wolf`	`"StringLike": { "s3:prefix": [ "home/", "home/nikki_wolf /" ] }`
`"StringLike": { "s3:prefix": [`	Tidak `aws:username` dalam konteks permintaan.	Tidak cocok

Kondisi kebijakan	Permintaan konteks	Hasil
`"home/", "home/${aws:username}/" ] }`		

Untuk contoh kebijakan yang menunjukkan cara menggunakan `Condition` elemen untuk membatasi akses ke sumber daya berdasarkan ID aplikasi dan ID pengguna untuk federasi OIDC, lihatAmazon S3: Memungkinkan pengguna Amazon Cognito mengakses objek di bucket mereka.

Operator kondisi string multinilai

Jika kunci dalam permintaan berisi beberapa nilai, operator string dapat memenuhi syarat dengan operator set `ForAllValues` dan`ForAnyValue`. Untuk informasi selengkapnya tentang logika evaluasi beberapa kunci konteks atau nilai, lihatMengatur operator untuk kunci konteks multinilai.

Operator ketentuan	Deskripsi

`ForAllValues:StringEquals`
 	
`ForAllValues:StringEqualsIgnoreCase`
	Semua nilai untuk kunci kondisi dalam permintaan harus cocok dengan setidaknya satu nilai dalam kebijakan Anda.

`ForAnyValue:StringEquals`
 	
`ForAnyValue:StringEqualsIgnoreCase`
	Setidaknya satu nilai kunci kondisi dalam permintaan harus cocok dengan salah satu nilai dalam kebijakan Anda.

`ForAllValues:StringNotEquals`
 	
`ForAllValues:StringNotEqualsIgnoreCase`
	Pencocokan yang dinegasikan.
	
	Tidak ada nilai kunci konteks dalam permintaan yang dapat mencocokkan nilai kunci konteks apa pun dalam kebijakan Anda.

`ForAnyValue:StringNotEquals`	Pencocokan yang dinegasikan. Setidaknya satu nilai kunci konteks dalam permintaan TIDAK boleh cocok dengan salah

Operator ketentuan	Deskripsi
`ForAnyValue:StringNotEquals IgnoreCase`	satu nilai dalam kunci konteks dalam kebijakan Anda.
`ForAllValues:StringLike`	Semua nilai untuk kunci kondisi dalam permintaan harus cocok dengan setidaknya satu nilai dalam kebijakan Anda.
`ForAnyValue:StringLike`	Setidaknya satu nilai kunci kondisi dalam permintaan harus cocok dengan salah satu nilai dalam kebijakan Anda.
`ForAllValues:StringNotLike`	Pencocokan yang dinegasikan. Tidak ada nilai kunci konteks dalam permintaa n yang dapat mencocokkan nilai kunci konteks apa pun dalam kebijakan Anda.
`ForAnyValue:StringNotLike`	Pencocokan yang dinegasikan. Setidaknya satu nilai kunci konteks dalam permintaan TIDAK boleh cocok dengan salah satu nilai dalam kunci konteks dalam kebijakan Anda.

Example menggunakan **`ForAnyValue`** dengan operator kondisi string

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan penggunaan `CreateTags` tindakan Amazon EC2 untuk melampirkan tag ke instans. Saat Anda menggunakan`StringEqualsIgnoreCase`, Anda dapat melampirkan tag hanya jika tag berisi `environment` kunci dengan `storage` nilai `preprod` atau. Saat Anda menambahkan `IgnoreCase` ke operator, Anda mengizinkan kapitalisasi nilai tag yang ada, seperti,`preprod`, dan `PreprodPreProd`, untuk diselesaikan menjadi true.

Saat Anda menambahkan `ForAnyValue` pengubah dengan kunci aws: TagKeys kondisi, setidaknya satu nilai kunci tag dalam permintaan harus cocok dengan nilainya. `environment` `ForAnyValue`perbandingan peka huruf besar/kecil, yang mencegah pengguna menggunakan huruf besar yang salah untuk kunci tag, seperti menggunakan al `Environment` ih-alih`environment`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:*:*:instance/*",
    "Condition": {
      "StringEqualsIgnoreCase": {
        "aws:RequestTag/environment": [
          "preprod",
          "storage"
        ]
      },
      "ForAnyValue:StringEquals": {
        "aws:TagKeys": "environment"
      }
    }
  }
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment"`	`aws:TagKeys: - environment aws:RequestTag/environment: - preprod`	Cocokkan

Kondisi kebijakan	Permintaan konteks	Hasil

}
"StringEqualsIgnor
eCase": {
"aws:RequestTag/en
vironment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEqu
als": {
"aws:TagKeys":
"environment"
}
	aws:TagKeys:
	- environment
	- costcenter
	aws:RequestTag/e
	nvironment:
	- PreProd
		Cocokkan

"StringEqualsIgnor
eCase": {
"aws:RequestTag/en
vironment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEqu
als": {
"aws:TagKeys":
"environment"
}
	aws:TagKeys:
	- Environment
	aws:RequestTag/
	Environment:
	- preprod
		Tidak cocok

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - costcenter aws:RequestTag/environment: - preprod`	Tidak cocok
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	Tidak `aws:TagKeys` dalam konteks permintaan. `aws:RequestTag/environment: - storage`	Tidak cocok

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment` Tidak `aws:RequestTag/environment` dalam konteks permintaan.	Tidak cocok
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	Tidak `aws:TagKeys` dalam konteks permintaan. Tidak `aws:RequestTag/environment` dalam konteks permintaan.	Tidak cocok

Pencocokan wildcard

Operator kondisi string melakukan pencocokan tanpa pola yang tidak menerapkan format yang telah ditentukan sebelumnya. Operator kondisi ARN dan Tanggal adalah subset dari operator string yang menerapkan struktur pada nilai kunci kondisi.

Sebaiknya gunakan operator kondisi yang sesuai dengan nilai yang Anda bandingkan dengan kunci. Misalnya, Anda harus menggunakan the section called “Operator ketentuan string” saat membandingkan kunci dengan nilai string. Demikian pula, Anda harus menggunakan the section

called “Operator ketentuan Amazon Resource Name (ARN)” ketika membandingkan kunci dengan nilai ARN.

Example

Contoh ini menunjukkan bagaimana Anda dapat membuat batas di sekitar sumber daya di organisasi Anda. Kondisi dalam kebijakan ini menolak akses ke tindakan Amazon S3 kecuali sumber daya yang diakses berada dalam kumpulan unit organisasi (OU) tertentu di AWS Organizations. Jal AWS Organizations ur adalah representasi teks dari struktur entitas organisasi.

Kondisi mengharuskan itu `aw: ResourceOrgPaths` berisi salah satu jalur OU yang terdaftar. Karena `aws:ResourceOrgPaths` merupakan kondisi multi-nilai, kebijakan menggunakan `ForAllValues:StringNotLike` operator untuk membandingkan nilai-nilai `aws:ResourceOrgPaths` dengan daftar OU dalam kebijakan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:ResourceOrgPaths": [
            "o-acorg/r-acroot/ou-acroot-mediaou/",
            "o-acorg/r-acroot/ou-acroot-sportsou/*"
          ]
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-sportsou/costcenter/`	Cocokkan
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-mediaou/costcenter/`	Tidak cocok
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	Tidak `aws:ResourceOrgPaths:` dalam permintaan.	Tidak cocok

Operator ketentuan numerik

Operator ketentuan numerik memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke nilai bilangan bulat atau desimal.

- Variabel kebijakan - Tidak didukung
- Karakter pengganti - Tidak didukung

Operator ketentuan	Deskripsi
`NumericEquals`	Pencocokan
`NumericNotEquals`	Pencocokan dinegasikan
`NumericLessThan`	Pencocokan “Kurang dari”
`NumericLessThanEqu als`	Pencocokan “kurang dari atau sama dengan”
`NumericGreaterThan`	Pencocokan “lebih besar dari”
`NumericGreaterThan Equals`	Pencocokan “lebih besar dari atau sama dengan”

Misalnya, pernyataan berikut memuat elemen `Condition` yang menggunakan operator ketentuan `NumericLessThanEquals` dengan kunci `s3:max-keys` untuk menentukan bahwa pemohon dapat membuat daftar hingga 10 objek dalam `amzn-s3-demo-bucket` sekali waktu.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
```

```
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Dalam contoh ini, kunci `s3:max-keys` selalu hadir dalam permintaan saat Anda melakukan operasi `ListBucket`. Jika kebijakan ini mengizinkan semua operasi Amazon S3 maka hanya operasi yang mencakup kunci konteks `max-keys` yang bernilai kurang dari atau sama dengan 10 akan diperbolehkan.

Operator ketentuan tanggal

Operator kondisi tanggal memungkinkan Anda membangun `Condition` elemen yang membatasi akses berdasarkan membandingkan kunci dengan date/time nilai. Anda menggunakan operator kondisi ini dengan `aws:CurrentTime` kunci atau `aws:EpochTime` kunci. Anda harus menentukan date/time nilai dengan salah satu implementasi W3C dari format tanggal ISO 8601 atau dalam waktu zaman (UNIX).

- Variabel kebijakan - Tidak didukung
- Karakter pengganti - Tidak didukung

Operator ketentuan	Deskripsi
`DateEquals`	Mencocokkan tanggal tertentu
`DateNotEquals`	Pencocokan dinegasikan
`DateLessThan`	Sesuai sebelum tanggal dan waktu tertentu
`DateLessThanEquals`	Sesuai pada atau sebelum tanggal dan waktu tertentu
`DateGreaterThan`	Sesuai setelah tanggal dan waktu tertentu
`DateGreaterThanEqu als`	Sesuai pada atau setelah tanggal dan waktu tertentu

Misalnya, pernyataan berikut memuat elemen `Condition` yang menggunakan operator ketentuan `DateGreaterThan` dengan kunci `aws:TokenIssueTime`. Ketentuan ini menyebutkan bahwa kredensial keamanan sementara yang digunakan untuk membuat permintaan tersebut diterbitkan

pada 2020. Kebijakan ini dapat secara terprogram diperbarui setiap hari untuk memastikan anggota akun menggunakan kredensial baru.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "DateGreaterThan": {
                "aws:TokenIssueTime": "2020-01-01T00:00:01Z"
            }
        }
    }
}
```

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Kunci `aws:TokenIssueTime` tersedia dalam konteks permintaan hanya ketika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kunci tidak ada dalam permintaan AWS CLI, AWS API, atau AWS SDK yang dibuat menggunakan kunci akses. Dalam contoh ini, jika pengguna IAM mencoba untuk melihat atau mengedit kunci akses, permintaan ditolak.

Operator ketentuan Boolean

Kondisi Boolean memungkinkan Anda membangun `Condition` elemen yang membatasi akses berdasarkan membandingkan kunci dengan atau. `true` `false`

Jika kunci berisi beberapa nilai, operator boolean dapat memenuhi syarat dengan operator `ForAllValues` set dan. `ForAnyValue` Untuk informasi selengkapnya tentang logika evaluasi beberapa kunci konteks atau nilai, lihatMengatur operator untuk kunci konteks multinilai.

- Variabel kebijakan - Di dukung
- Karakter pengganti - Tidak didukung

Operator ketentuan	Deskripsi
`Bool`	Pencocokan Boolean
`ForAllValues:Bool`	Gunakan dengan tipe data Array of Bool. Semua boolean dalam nilai kunci konteks harus sesuai dengan nilai boolean dalam kebijakan Anda. Untuk mencegah `ForAllValues` operator mengevaluasi kunci konteks yang hilang atau kunci konteks dengan nilai kosong sebagai Diizinkan, Anda dapat menyertakan operator kondisi `Null` dalam kebijakan Anda.
`ForAnyValue:Bool`	Gunakan dengan tipe data Array of Bool. Setidaknya satu boolean dalam nilai kunci konteks harus cocok dengan nilai boolean dalam kebijakan Anda.

Example operator kondisi boolean

Kebijakan berbasis identitas berikut menggunakan operator `Bool` kondisi dengan `aws:SecureTransport` kunci untuk menolak mereplikasi objek dan tag objek ke bucket tujuan dan isinya jika permintaan tidak melewati SSL.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
```

```
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - false`	Cocokkan
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - true`	Tidak cocok
`"Bool": { "aws:SecureTranspo rt": "false" }`	Tidak `aws:Secur eTransport` dalam konteks permintaan.	Tidak cocok

Operator ketentuan biner

Operator `BinaryEquals` kondisi memungkinkan Anda membangun `Condition` elemen yang menguji nilai kunci yang dalam format biner. Ini membandingkan nilai dari byte kunci yang ditentukan

untuk byte terhadap perwakilan terkode base-64 dari nilai biner di dalam kebijakan. Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok.

- Variabel kebijakan - Tidak didukung
- Karakter pengganti - Tidak didukung

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Kondisi kebijakan	Permintaan konteks	Hasil
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - QmluYXJ5VmFsdWVJbk Jhc2U2NA==`	Cocokkan
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - ASIAIOSFODNN7EXAMP LE`	Tidak cocok
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	Tidak `key` dalam konteks permintaan.	Tidak cocok

Operator ketentuan alamat IP

Operator kondisi alamat IP memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke alamat IPv4 atau IPv6 atau rentang alamat IP. Anda menggunakan ini dengan kunci `aws:SourceIp`. Nilai harus dalam format CIDR standar (misalnya, 203.0.113. 0/24 atau 2001:DB8:1234:5678: :/64). Jika Anda menentukan alamat IP tanpa prefiks perutean yang terkait, IAM menggunakan nilai prefiks default dari `/32`.

Beberapa AWS layanan mendukung IPv6, menggunakan:: untuk mewakili rentang 0s. Untuk mempelajari apakah layanan mendukung IPv6, lihat dokumentasi untuk layanan tersebut.

- Variabel kebijakan - Tidak didukung
- Karakter pengganti - Tidak didukung

Operator ketentuan	Deskripsi
`IpAddress`	Alamat IP atau rentang yang ditentukan
`NotIpAddress`	Semua alamat IP kecuali alamat dan rentang IP yang ditentukan

Example Operator kondisi alamat IP

Pernyataan berikut menggunakan operator `IpAddress` kondisi dengan `aws:SourceIp` kunci untuk menentukan bahwa permintaan harus berasal dari kisaran IP 203.0.113.0 hingga 203.0.113.255.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "IpAddress": {
                "aws:SourceIp": "203.0.113.0/24"
            }
        }
    }
}
```

Kunci kondisi `aws:SourceIp` menyelesaikan ke alamat IP yang berasal dari permintaan asli. Jika permintaan berasal dari instans Amazon EC2, `aws:SourceIp` mengevaluasi alamat IP publik instans.

Jika kunci yang Anda sebutkan dalam ketentuan kebijakan tidak ada dalam konteks permintaan, nilainya tidak cocok. Kunci `aws:SourceIp` selalu ada dalam konteks permintaan, kecuali saat pemohon menggunakan VPC endpoint untuk membuat permintaan. Dalam hal ini, ketentuan mengembalikan `false` dan permintaan tersebut ditolak secara tersirat oleh pernyataan ini.

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 203.0.113.1`	Cocokkan
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 198.51.100.1`	Tidak cocok

Contoh berikut menunjukkan cara menggabungkan rentang alamat IPv4 dan IPv6 untuk mencakup semua alamat IP yang valid dalam organisasi Anda. Kami menyarankan Anda memperbarui kebijakan organisasi Anda dengan rentang alamat IPv6 Anda selain rentang IPv4 Anda sudah harus memastikan kebijakan terus berfungsi saat Anda melakukan transisi ke IPv6.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
```

```
        ]
      }
    }
  }
}
```

Kunci kondisi `aws:SourceIp` hanya berfungsi dalam kebijakan JSON jika Anda menghubungi API yang diuji secara langsung sebagai pengguna. Jika Anda menggunakan layanan untuk menelepon layanan target atas nama Anda, layanan target akan melihat alamat IP layanan panggilan, bukan alamat IP pengguna asal. Hal ini dapat terjadi, misalnya, jika Anda menggunakan AWS CloudFormation untuk memanggil Amazon EC2 untuk membuat instans untuk Anda. Saat ini tidak ada cara untuk menyampaikan alamat IP asal melalui layanan panggilan ke layanan target untuk evaluasi di kebijakan JSON. Untuk tipe panggilan API layanan ini, jangan gunakan kunci kondisi `aws:SourceIp`.

Operator ketentuan Amazon Resource Name (ARN)

Operator ketentuan Amazon Resource Name (ARN) memungkinkan Anda membangun elemen `Condition` yang membatasi akses berdasarkan perbandingan kunci ke ARN. ARN dianggap sebagai string.

- Variabel kebijakan - Di dukung
- Wildcard - Didukung ???

Operator ketentuan	Deskripsi
`ArnEquals` , `ArnLike`	Case-sensitive pencocokan ARN. Masing-masing dari enam komponen ARN yang dibatasi titik dua diperiksa secara terpisah dan masing-masing dapat menyertakan wildcard kecocokan multi-karakter (*) atau wildcard pencocokan karakter tunggal (?). Oper `ArnEquals` ator `ArnLike` kondisi dan berperilaku identik.
`ArnNotEquals` , `ArnNotLike`	Pencocokan negasi untuk ARN. Oper `ArnNotEquals` ator `ArnNotLike` kondisi dan berperilaku identik.

Example Operator kondisi ARN

Contoh kebijakan berbasis sumber daya berikut ini menunjukkan kebijakan yang terlampir pada antrean Amazon SQS di mana Anda ingin mengirimkan pesan SNS. Ini memberi izin Amazon SNS untuk mengirimkan pesan ke antrean (atau antrean-antrean) pilihan Anda, tetapi hanya jika layanan tersebut mengirimkan pesan atas nama topik (atau topik) Amazon SNS tertentu. Anda menentukan antrean di bidang `Resource`, dan topik Amazon SNS sebagai nilai untuk kunci `SourceArn`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "sns.amazonaws.com"
        },
        "Action": "SQS:SendMessage",
        "Resource": "arn:aws:sqs:us-east-1:123456789012:QUEUE-ID",
        "Condition": {
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:sns:us-east-1:123456789012:TOPIC-ID"
            }
        }
    }
}
```

Kunci `aws:SourceArn` muncul dalam konteks permintaan hanya jika sumber daya memicu layanan untuk menghubungi layanan lain atas nama pemilik sumber daya. Jika pengguna IAM mencoba melakukan operasi ini secara langsung, ketentuan akan kembali `false` dan permintaan tersebut ditolak secara tersirat oleh pernyataan ini.

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"ArnEquals": {`	`aws:SourceArn:`	Cocokkan

Kondisi kebijakan	Permintaan konteks	Hasil
`"aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`- arn:aws:sns:us-west-2:123456789012:TOPIC-ID`	
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:777788889999:TOPIC-ID`	Tidak cocok
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	Tidak `aws:SourceArn` dalam konteks permintaan.	Tidak cocok

Operator kondisi ARN multinilai

Jika kunci dalam permintaan berisi beberapa nilai, operator ARN dapat memenuhi syarat dengan operator set `ForAllValues` dan`ForAnyValue`. Untuk informasi selengkapnya tentang logika evaluasi beberapa kunci konteks atau nilai, lihatMengatur operator untuk kunci konteks multinilai.

Operator ketentuan	Deskripsi
`ForAllValues:ArnEquals` `ForAllValues:ArnLike`	Semua ARN dalam konteks permintaan harus cocok dengan setidaknya satu pola ARN dalam kebijakan Anda.
`ForAnyValue:ArnEquals` `ForAnyValue:ArnLike`	Setidaknya satu ARN dalam konteks permintaan harus cocok dengan salah satu pola ARN dalam kebijakan Anda.

Operator ketentuan	Deskripsi

`ForAllValues:ArnNotEquals`
 	
`ForAllValues:ArnNotLike`
	Pencocokan yang dinegasikan.
	
	Tak satu pun dari ARN dalam konteks permintaan dapat mencocokkan pola ARN string apa pun dalam kebijakan Anda.

`ForAnyValue:ArnNotEquals`
 	
`ForAnyValue:ArnNotLike`
	Pencocokan yang dinegasikan.
	
	Setidaknya satu ARN dalam konteks permintaa n TIDAK boleh cocok dengan pola ARN apa pun dalam kebijakan Anda.

Example menggunakan **`ForAllValues`** dengan operator kondisi ARN

Contoh berikut digunakan `ForAllValues:ArnLike` untuk membuat atau memperbarui sumber pengiriman logis untuk CloudWatch log Amazon Logs. Blok kondisi menyertakan kunci kondisi `logs:LogGeneratingResourceArns` untuk memfilter ARN sumber daya penghasil log yang diteruskan dalam permintaan. Menggunakan operator kondisi ini, semua ARN dalam permintaan harus cocok dengan setidaknya satu ARN dalam kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "logs:PutDeliverySource",
            "Resource": "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
            "Condition": {
                "ForAllValues:ArnLike": {
                    "logs:LogGeneratingResourceArns": [
                        "arn:aws:cloudfront::123456789012:distribution/*",
                        "arn:aws:cloudfront::123456789012:distribution/support*"
                    ]
                }
            }
```

```
        }
    ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter`	Cocokkan
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/support2025`	Cocokkan
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [`	`logs:LogGeneratingResourceArns:`	Cocokkan

Kondisi kebijakan	Permintaan konteks	Hasil
`"arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`- arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/admin`	
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:777788889999:distribution/costcenter`	Tidak cocok
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	Tidak `logs:LogGeneratingResourceArns` dalam konteks permintaan.	Cocokkan

`ForAllValues`Qualifier mengembalikan true jika tidak ada kunci konteks dalam permintaan atau jika nilai kunci konteks diselesaikan ke dataset null, seperti string kosong. Untuk mencegah kunci konteks yang hilang atau kunci konteks dengan nilai kosong dievaluasi menjadi true, Anda dapat

menyertakan operator kondisi Null dalam kebijakan Anda dengan `false` nilai untuk memeriksa apakah kunci konteks ada dan nilainya tidak null.

... IfExists operator kondisi

Anda dapat menambahkan `IfExists` ke akhir nama operator kondisi apa pun kecuali kondisi `Null` —misalnya,. `StringLikeIfExists` Anda melakukan ini untuk mengatakan “Jika kunci kondisi ada dalam konteks permintaan, proses kunci seperti yang ditentukan dalam kebijakan. Jika kuncinya tidak ada, evaluasi elemen ketentuan sebagai benar." Elemen ketentuan lain dalam pernyataan ini masih dapat menghasilkan ketidakcocokkan, tetapi bukan kunci yang hilang saat diperiksa dengan `...IfExists`. Jika Anda menggunakan `"Effect": "Deny"` elemen dengan operator kondisi dinegasikan sepertiStringNotEqualsIfExists, permintaan masih ditolak meskipun kunci kondisi tidak ada.

Contoh menggunakan **`IfExists`**

Banyak kunci kondisi menjelaskan informasi tentang tipe sumber daya tertentu dan hanya ada ketika mengakses tipe sumber daya tersebut. Kunci kepatuhan ini tidak ada pada tipe sumber daya lainnya. Hal ini tidak menyebabkan masalah ketika pernyataan kebijakan berlaku hanya pada satu tipe sumber daya. Namun, ada kasus ketika pernyataan tunggal dapat berlaku untuk beberapa tipe sumber daya, seperti ketika pernyataan kebijakan merujuk tindakan dari beberapa layanan atau ketika tindakan tertentu dalam layanan mengakses beberapa tipe sumber daya yang berbeda dalam layanan yang sama. Dalam kasus seperti itu, termasuk kunci kepatuhan yang berlaku hanya pada salah satu sumber daya dalam pernyataan kebijakan dapat menyebabkan elemen `Condition` dalam pernyataan kebijakan gagal sehingga pernyataan `"Effect"` tidak berlaku.

Misalnya, pertimbangkan contoh kebijakan berikut ini:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
```

```
      "m3.*"
    ]}}
  }
}
```

Mak sud dari kebijakan sebelumnya adalah untuk memungkinkan pengguna meluncurkan instance apa pun yang berjenist1, t2 ataum3. Namun, meluncurkan instance memerlukan akses ke banyak sumber daya selain instance itu sendiri; misalnya, gambar, pasangan kunci, grup keamanan, dan banyak lagi. Keseluruhan pernyataan dievaluasi terhadap setiap sumber daya yang diperlukan untuk meluncurkan instans. Sumber daya tambahan ini tidak memiliki kunci kondisi ec2:InstanceType, jadi StringLike gagal diperiksa, dan pengguna tidak diberi kemampuan untuk meluncurkan tipe instans mana pun.

Untuk mengatasi ini, gunakan operator ketentuan StringLikeIfExists. Dengan cara ini, pengujian hanya terjadi jika terdapat kunci kondisi. Anda dapat membaca kebijakan berikut sebagai: “Jika sumber daya yang diperiksa memiliki kunci kondisi, maka izinkan tindakan hanya jika nilai kunci dimulai dengant1.,t2., ataum3.. ec2:InstanceType Jika sumber daya yang diperiksa tidak memiliki kunci kondisi itu, jangan khawatir tentang hal tersebut.” Tanda bintang (*) dalam nilai kunci kondisi, bila digunakan dengan operator StringLikeIfExists kondisi, ditafsirkan sebagai wildcard untuk mencapai kecocokan string parSIAL. Pernyataan DescribeActions mencakup tindakan yang diperlukan untuk melihat instans di konsol.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RunInstance",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
        "StringLikeIfExists": {
          "ec2:InstanceType": [
            "t1.*",
            "t2.*",
            "m3.*"
          ]
```

```
        }
      }
    },
    {
      "Sid": "DescribeActions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeImages",
        "ec2:DescribeInstances",
        "ec2:DescribeVpcs",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - t1.micro`	Cocokkan
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*",`	`ec2:InstanceType: - m2.micro`	Tidak cocok

Kondisi kebijakan	Permintaan konteks	Hasil

`"m3.*"`
`]`
`}`		

`"StringLikeIfExists":`
`{`
`"ec2:InstanceType":`
`[`
`"t1.*",`
`"t2.*",`
`"m3.*"`
`]`
`}`
	Tidak `ec2:InstanceType` dalam konteks permintaan.	Cocokkan

Operator ketentuan memeriksa keberadaan kunci kondisi

Gunakan operator `Null` kondisi untuk memeriksa apakah kunci kondisi tidak ada pada saat otorisasi. Dalam pernyataan kebijakan, gunakan `true` (kuncinya tidak ada — yaitu null) atau `false` (kunci ada dan nilainya tidak null).

Anda tidak dapat menggunakan variabel kebijakan dengan operator ketentuan `Null`.

Misalnya, Anda dapat menggunakan operator kondisi ini untuk menentukan apakah pengguna menggunakan kredentif sementara atau kredensialnya sendiri untuk membuat permintaan. Jika pengguna menggunakan kredensial sementara, maka kuncinya `aws:TokenIssueTime` ada dan memiliki nilai. Contoh berikut menunjukkan kondisi yang menyatakan bahwa pengguna harus menggunakan kredenSIAL sementara (kunci tidak boleh absen) agar pengguna dapat menggunakan Amazon EC2 API.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
```

```
      "Condition":{"Null":{"aws:TokenIssueTime":"false"}}
  }
}
```

Kondisi dengan beberapa kunci konteks atau nilai

Anda dapat menggunakan `Condition` elemen kebijakan untuk menguji beberapa kunci konteks atau beberapa nilai untuk satu kunci konteks dalam permintaan. Saat Anda mengajukan permintaan AWS, baik secara terprogram atau melalui Konsol Manajemen AWS, permintaan Anda mencakup informasi tentang prinsipal, operasi, tag, dan lainnya. Anda dapat menggunakan kunci konteks untuk menguji nilai kunci konteks yang cocok dalam permintaan, dengan kunci konteks yang ditentukan dalam kondisi kebijakan. Untuk mempelajari informasi dan data yang disertakan dalam permintaan, lihat Konteks permintaan.

Topik

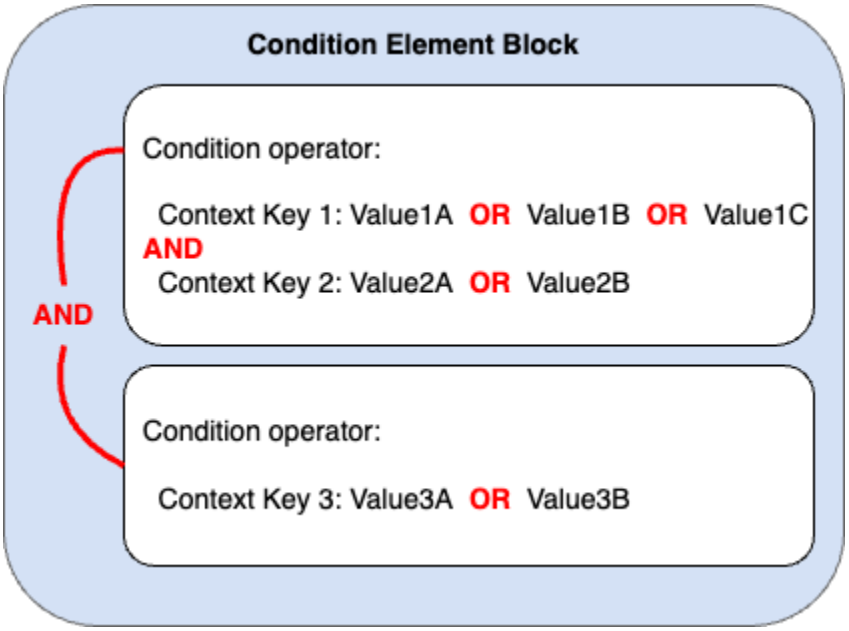
- Logika evaluasi untuk beberapa kunci konteks atau nilai
- Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan

Logika evaluasi untuk beberapa kunci konteks atau nilai

Sebuah `Condition` elemen dapat berisi beberapa operator kondisi, dan setiap operator kondisi dapat berisi beberapa pasangan kunci-nilai konteks. Sebagian besar kunci konteks mendukung penggunaan beberapa nilai, kecuali ditentukan lain.

- Jika pernyataan kebijakan Anda memiliki beberapa operator kondisi, operator kondisi dievaluasi menggunakan logika`AND`.
- Jika pernyataan kebijakan Anda memiliki beberapa kunci konteks yang dilampirkan ke satu operator kondisi, kunci konteks dievaluasi menggunakan logika`AND`.
- Jika operator kondisi tunggal menyertakan beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logika`OR`.
- Jika operator kondisi pencocokan dinegasikan tunggal menyertakan beberapa nilai untuk kunci konteks, nilai-nilai tersebut dievaluasi menggunakan logika. `NOR`

Semua kunci konteks dalam blok elemen kondisi harus diselesaikan ke true untuk memanggil yang diinginkan `Allow` atau `Deny` efek. Gambar berikut mengilustrasikan logika evaluasi untuk suatu kondisi dengan beberapa operator kondisi dan pasangan kunci-nilai konteks.



Misalnya, kebijakan bucket S3 berikut menggambarkan bagaimana gambar sebelumnya direpresentasikan dalam kebijakan. Blok kondisi mencakup operator kondisi `StringEquals` dan`ArnLike`, dan kunci konteks `aws:PrincipalTag` dan`aws:PrincipalArn`. Untuk memanggil yang diinginkan `Allow` atau `Deny` efek, semua kunci konteks di blok kondisi harus diselesaikan ke true. Pengguna yang membuat permintaan harus memiliki kunci tag utama, departemen, dan peran, yang menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan. Selain itu, ARN utama pengguna yang membuat permintaan harus cocok dengan salah satu `aws:PrincipalArn` nilai yang ditentukan dalam kebijakan untuk dievaluasi sebagai benar.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
```

```
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`	Pertandingan

Kondisi Kebijakan	Permintaan Konteks	Hasil

`"arn:aws:iam::222222222222:user/Ana",`
`"arn:aws:iam::222222222222:user/Mary"`
`]`
`}`		

`"StringEquals": {`
`"aws:PrincipalTag/department": [`
`"finance",`
`"hr",`
`"legal"`
`],`
`"aws:PrincipalTag/role": [`
`"audit",`
`"security"`
`]`
`},`
`"ArnLike": {`
`"aws:PrincipalArn": [`
`"arn:aws:iam::222222222222:user/Ana",`
`"arn:aws:iam::222222222222:user/Mary"`
`]`
`}`
	`aws:PrincipalTag/department: hr`
	`aws:PrincipalTag/role: audit`
	`aws:PrincipalArn: arn:aws:iam::222222222222:user/ Nikki`
		Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: payroll aws:PrincipalArn: arn:aws:iam::22222222222222:user/Mary`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil

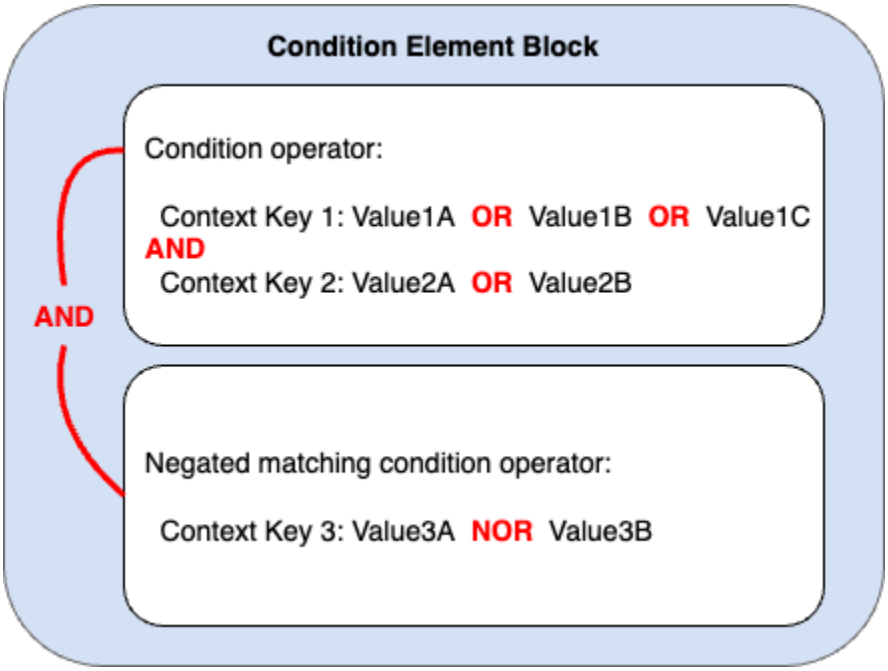
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Tidak `aws:PrincipalTag/role` dalam konteks permintaan.
	`aws:PrincipalTag/department: hr`
	`aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`
		Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Tidak `aws:PrincipalTag` dalam konteks permintaan. `aws:PrincipalArn: arn:aws:iam::22222222222222:user/Mary`	Tidak cocok

Logika evaluasi untuk operator kondisi pencocokan yang dinegasikan

Beberapa operator kondisi, seperti `StringNotEquals` atau`ArnNotLike`, menggunakan pencocokan yang dinegasikan untuk membandingkan pasangan nilai kunci konteks dalam kebijakan Anda dengan pasangan nilai kunci konteks dalam permintaan. Ketika beberapa nilai ditentukan untuk satu kunci konteks dalam kebijakan dengan operator kondisi pencocokan yang dinegasikan, izin efektif berfungsi seperti logika. `NOR` Dalam pencocokan yang dinegasikan, logis `NOR` atau `NOT OR` mengembalikan nilai true hanya jika semua nilai dievaluasi menjadi false.

Gambar berikut mengilustrasikan logika evaluasi untuk suatu kondisi dengan beberapa operator kondisi dan pasangan kunci-nilai konteks. Angka tersebut mencakup operator kondisi pencocokan yang dinegasikan untuk kunci konteks 3.



Misalnya, kebijakan bucket S3 berikut menggambarkan bagaimana gambar sebelumnya direpresentasikan dalam kebijakan. Blok kondisi mencakup operator kondisi `StringEquals` dan`ArnNotLike`, dan kunci konteks `aws:PrincipalTag` dan`aws:PrincipalArn`. Untuk memanggil yang diinginkan `Allow` atau `Deny` efek, semua kunci konteks di blok kondisi harus diselesaikan ke true. Pengguna yang membuat permintaan harus memiliki kunci tag utama, departemen, dan peran, yang menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan. Karena operator `ArnNotLike` kondisi menggunakan pencocokan yang dinegasikan, ARN utama pengguna yang membuat permintaan tidak boleh cocok dengan nilai yang `aws:PrincipalArn` ditentukan dalam kebijakan yang akan dievaluasi sebagai true.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] },`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Pertandingan

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`		
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/ Mary`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: payroll aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	**>** Tidak `aws:PrincipalTag/role` dalam konteks permintaan. `aws:PrincipalTag/department: hr aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Tidak `aws:PrincipalTag` dalam konteks permintaan. `aws:PrincipalArn: arn:aws:iam::22222222222222:user/Nikki`	Tidak cocok

Single-valued vs. kunci konteks multinilai

Perbedaan antara kunci konteks bernilai tunggal dan multinilai terletak pada jumlah nilai dalam konteks permintaan, bukan jumlah nilai dalam kondisi kebijakan.

- Single-valuedkunci konteks kondisi memiliki paling banyak satu nilai dalam konteks permintaan. Misalnya, saat Anda menandai resource AWS, setiap tag resource disimpan sebagai pasangan kunci-nilai. Karena kunci tag sumber daya hanya dapat memiliki nilai tag tunggal, the section called “ResourceTag” adalah kunci konteks bernilai tunggal. Jangan gunakan operator set kondisi dengan kunci konteks bernilai tunggal.
- Kunci konteks kondisi multivalue dapat memiliki beberapa nilai dalam konteks permintaan. Misalnya, saat Anda menandai sumber daya AWS, Anda dapat menyertakan beberapa pasangan

kunci-nilai tag dalam satu permintaan. Oleh karena itu, the section called “TagKeys” adalah kunci konteks multinilai. Kunci konteks multinilai memerlukan operator set kondisi.

Misalnya, permintaan dapat berasal dari paling banyak satu titik akhir VPC, begitu the section called “SourceVpce” juga kunci konteks bernilai tunggal. Karena layanan dapat memiliki lebih dari satu nama utama layanan yang termasuk dalam layanan, the section called “PrincipalServiceNamesList” adalah kunci konteks multinilai.

> **Important**
>
> Perbedaan antara kunci konteks bernilai tunggal dan multinilai bergantung pada jumlah nilai dalam konteks permintaan, bukan jumlah nilai dalam kondisi kebijakan.

Untuk contoh kebijakan yang menunjukkan kondisi dengan beberapa kunci konteks dan nilai, lihatContoh kebijakan kondisi.

Poin kunci

- Klasifikasi Single-valued dan Multivalue disertakan dalam deskripsi setiap kunci konteks kondisi sebagai tipe Nilai dalam topik. AWS kunci konteks kondisi global
- Kunci konteks multinilai dalam Refer ensi Otorisasi Layanan menggunakan `ArrayOf` awalan yang diikuti oleh tipe kategori operator kondisi, seperti `ArrayOfString` atau`ArrayOfARN`, yang menunjukkan bahwa permintaan dapat menyertakan beberapa nilai untuk kunci konteks kondisi.
- Anda dapat menggunakan kunci konteks bernilai tunggal yang tersedia sebagai variabel kebijakan, tetapi Anda tidak dapat menggunakan kunci konteks multinilai sebagai variabel kebijakan. Untuk informasi lebih lanjut tentang variabel-variable kebijakan, lihat Elemen kebijakan IAM: Variabel dan tag.
- Saat menggunakan kunci konteks yang menyertakan pasangan kunci-nilai, penting untuk dicatat bahwa meskipun mungkin ada beberapa nilai tag-key, masing-masing hanya `tag-key` dapat memiliki satu nilai.
  - the section called “PrincipalTag”, the section called “RequestTag” dan the section called “ResourceTag” merupakan kunci konteks bernilai tunggal.
  - the section called “TagKeys”mendefinisikan kunci tag apa yang diizinkan dalam permintaan tetapi tidak menyertakan nilai tag-key. Karena Anda dapat menyertakan beberapa pasangan kunci-nilai tag dalam permintaan, `aws:TagKeys` adalah kunci konteks multinilai.

- Kunci konteks multinilai memerlukan operator set kondisi. Jangan gunakan operator set kondisi `ForAllValues` atau `ForAnyValue` dengan kunci konteks bernilai tunggal. Menggunakan operator set kondisi dengan kunci konteks bernilai tunggal dapat menyebabkan kebijakan yang terlalu permisif.

Mengatur operator untuk kunci konteks multinilai

Untuk membandingkan kunci konteks kondisi Anda dengan kunci konteks permintaan dengan beberapa nilai, Anda harus menggunakan operator `ForAllValues` atau `ForAnyValue` set. Operator set ini digunakan untuk membandingkan dua set nilai, seperti set tag dalam permintaan dan kumpulan tag dalam kondisi kebijakan.

`ForAnyValue`Qualifier `ForAllValues` and menambahkan fungsionalitas pengaturan operasi ke operator kondisi, memungkinkan Anda menguji kunci konteks permintaan dengan beberapa nilai terhadap beberapa nilai kunci konteks dalam kondisi kebijakan. Selain itu, jika Anda menyertakan kunci konteks string multinilai dalam kebijakan Anda dengan wildcard atau variabel, Anda juga harus menggunakan operator `StringLike` kondisi. Beberapa nilai kunci kondisi harus ditutup dalam tanda kurung seperti array, misalnya,`"Key2":["Value2A", "Value2B"]`.

ForAllValues

`ForAllValues`Qualifier menguji apakah nilai setiap anggota konteks permintaan cocok dengan operator kondisi yang mengikuti kualifikasi. Kondisi dikembalikan `true` jika setiap nilai kunci konteks dalam permintaan cocok dengan nilai kunci konteks dalam kebijakan. Ini juga kembali `true` jika tidak ada kunci konteks dalam permintaan.

> **Important**
>
> Berhati-hatilah jika Anda menggunakan `ForAllValues` dengan `Allow` efek, karena bisa terlalu permisif jika kehadiran kunci konteks yang hilang dalam konteks permintaan tidak terduga. Anda harus selalu menyertakan operator `Null` kondisi dalam kebijakan Anda dengan `false` nilai untuk memeriksa apakah kunci konteks ada dan nilainya tidak null. Sebagai contoh, lihat Mengontrol akses berdasarkan kunci tanda.

Contoh ForAllValues set operator

Dalam contoh berikut, ForAllValues digunakan dengan aws: TagKeys untuk memungkinkan pengguna menghapus tag tertentu yang ditetapkan ke instance EC2. Kebijakan ini memungkinkan

pengguna untuk menghapus hanya `cost-center` tag `environment` dan. Anda dapat menghapusnya secara terpisah atau bersama-sama. Kunci tag dalam permintaan harus sama persis dengan kunci yang ditentukan dalam kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                },
                "Null": {
                    "aws:TagKeys": "false"
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] },`	`aws:TagKeys: - environment`	Cocokkan

Kondisi Kebijakan	Permintaan Konteks	Hasil

"Null": {
"aws:TagKeys":
"false"
}		

"ForAllValues:Stri
ngEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}
	aws:TagKeys:
	- cost-center
		Cocokkan

"ForAllValues:Stri
ngEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}
	aws:TagKeys:
	- environment
	- cost-center
		Cocokkan

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment - dept`	Tidak cocok
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	Tidak `aws:TagKeys` dalam konteks permintaan.	Tidak cocok

Perhatikan bahwa dalam contoh terakhir, hasilnya adalah “Tidak cocok” karena pemeriksaan kondisi Null mencegah pencocokan saat kunci konteks hilang. Ini adalah praktik terbaik untuk menghindari kebijakan yang terlalu permisif.

ForAnyValue

`ForAnyValue`Qualifier menguji apakah setidaknya satu anggota dari kumpulan nilai kunci konteks permintaan cocok dengan setidaknya satu anggota dari kumpulan nilai kunci konteks dalam kondisi kebijakan Anda. Kondisi dikembalikan `true` jika salah satu nilai kunci konteks dalam permintaan cocok dengan salah satu nilai kunci konteks dalam kebijakan. Untuk tidak ada kunci konteks yang cocok atau jika kunci tidak ada, kondisi kembali`false`.

> **Important**
>
> Saat menggunakan `ForAnyValue` dengan `Deny` efek, jika kunci konteks tidak ada dalam permintaan, kebijakan akan dinilai sebagai Tidak cocok. Untuk perilaku yang konsisten, tambahkan pemeriksaan `Null` kondisi eksplisit di kebijakan Anda untuk memverifikasi apakah kunci konteks ada. Lihat perinciannya di Operator ketentuan memeriksa keberadaan kunci kondisi.

Contoh ForAnyValue set operator

Dalam contoh berikut, ForAnyValue digunakan dengan aws: TagKeys untuk memungkinkan pengguna menghapus tag tertentu yang ditetapkan ke instance EC2. Kebijakan ini memungkinkan pengguna untuk menghapus tag untuk instance jika kunci tag yang ditentukan dalam permintaan termasuk `environment` atau`cost-center`. Permintaan dapat menyertakan kunci tag tambahan di luar yang ditentukan dalam kebijakan, tetapi harus menyertakan setidaknya satu kunci yang ditentukan agar sesuai dengan kondisi tersebut.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                }
            }
        }
    ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment`	Cocokkan
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - cost-center`	Cocokkan
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - cost-center`	Cocokkan
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - dept`	Cocokkan
`"ForAnyValue:StringEquals": {`	`aws:TagKeys:`	Tidak cocok

Kondisi Kebijakan	Permintaan Konteks	Hasil

<pre>"aws:TagKeys": [
"environment",
"cost-center"
]
}</pre>
	<pre>- dept</pre>	

<pre>"ForAnyValue:StringEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
}</pre>
	Tidak `aws:TagKeys` dalam konteks permintaan.	Tidak cocok

Contoh kebijakan kondisi

Dalam kebijakan IAM, Anda dapat menentukan beberapa nilai untuk kunci konteks bernilai tunggal dan multinilai untuk perbandingan dengan konteks permintaan. Kumpulan contoh kebijakan berikut menunjukkan kondisi kebijakan dengan beberapa kunci konteks dan nilai.

> **Note**
>
> Jika Anda ingin memberikan sebuah kebijakan untuk disertakan dalam panduan referensi ini, gunakan tombol Umpan Balik di bagian bawah halaman ini. Untuk contoh kebijakan berbasis identitas IAM, lihat. Contoh kebijakan berbasis identitas IAM

Contoh kebijakan kondisi: kunci Single-valued konteks

- Beberapa blok kondisi dengan kunci konteks bernilai tunggal
- Satu blok kondisi dengan beberapa kunci konteks nilai tunggal dan nilai

Contoh kebijakan kondisi: Kunci konteks multi-nilai

- Kebijakan tolak dengan operator set kondisi ForAllValues
- Kebijakan tolak dengan operator set kondisi ForAnyValue

Contoh kunci konteks multivaluasi

Kumpulan contoh kebijakan berikut menunjukkan cara membuat kondisi kebijakan dengan kunci konteks multivalued.

Contoh: Tolak kebijakan dengan operator set kondisi ForAllValues

Contoh berikut menunjukkan cara menggunakan kebijakan berbasis identitas untuk menolak penggunaan tindakan penandaan IAM saat awalan kunci tag tertentu disertakan dalam permintaan. Nilai untuk `aws:TagKeys`menyertakan wildcard (*) untuk pencocokan string paral. Kebijakan ini menyertakan operator yang `ForAllValues` disetel dengan kunci konteks `aws:TagKeys` karena kunci konteks permintaan dapat menyertakan beberapa nilai. Agar kunci `aws:TagKeys` konteks cocok, setiap nilai dalam konteks permintaan harus cocok dengan setidaknya satu nilai dalam kebijakan.

Operator `ForAllValues` set juga mengembalikan true jika tidak ada kunci konteks dalam permintaan.

Anda dapat mencegah kunci konteks atau kunci konteks yang hilang dengan nilai kosong dievaluasi ke true dengan menyertakan operator `Null` kondisi dalam kebijakan Anda dengan nilai `false` untuk memeriksa apakah kunci konteks dalam permintaan ada dan nilainya bukan null. Untuk informasi selengkapnya, lihat Operator ketentuan memeriksa keberadaan kunci kondisi.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

Example Tolak nilai kondisi kebijakan tunggal untuk kunci konteks multivalued

Dalam contoh berikut, kebijakan menolak permintaan di mana nilai untuk **`aws:TagKeys`** dalam permintaan tidak menyertakan kunci awalan1. Konteks permintaan dapat memiliki beberapa nilai, tetapi karena operator set `ForAllValues` kondisi, semua nilai kunci tag dalam konteks permintaan harus dimulai dengan kunci awalan1.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": "key1*"
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda. Untuk pernyataan Deny, Pertandingan Ditolak dan Tidak ada kecocokan Tidak ditolak, jadi mungkin diizinkan oleh pernyataan lain.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:legal`	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:hr - key1:personnel`	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key2:audit`	Pertandingan
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	Tidak `aws:TagKeys` dalam konteks permintaan.	Pertandingan

Example Tolak beberapa nilai kondisi kebijakan untuk kunci konteks multivalued

Dalam contoh berikut, kebijakan menolak permintaan di mana nilai untuk **`aws:TagKeys`** dalam permintaan tidak menyertakan awalan key1 atau key2. Konteks permintaan dapat memiliki beberapa nilai, tetapi karena operator set **`ForAllValues`** kondisi, semua nilai kunci tag dalam konteks permintaan harus dimulai dengan awalan key1 atau key2.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": [
```

```
            "key1*",
            "key2*"
          ]
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda. Untuk pernyataan Deny, Pertandingan Ditolak dan Tidak ada kecocokan Tidak ditolak, jadi mungkin diizinkan oleh pernyataan lain.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:legal`	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key1:personnel`	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key2:audit`	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key3:legal`	Pertandingan
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	Tidak `aws:TagKeys` dalam konteks permintaan.	Pertandingan

Contoh: Tolak kebijakan dengan operator set kondisi ForAnyValue

Contoh kebijakan berbasis identitas berikut menyangkal pembuatan snapshot volume instans EC2 jika ada snapshot yang ditandai dengan salah satu kunci tag yang ditentukan dalam kebijakan, atau. `environment` `webserver` Kebijakan ini menyertakan operator yang `ForAnyValue` disetel dengan kunci konteks `aws:TagKeys` karena kunci konteks permintaan dapat menyertakan beberapa nilai. Jika permintaan penandaan Anda menyertakan salah satu nilai kunci tag yang ditentukan dalam kebijakan, kunci `aws:TagKeys` konteks akan mengembalikan nilai true dengan mengaktifkan efek kebijakan penolakan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "ec2:CreateSnapshot",
        "ec2:CreateSnapshots"
      ],
      "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "webserver"
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda. Untuk pernyataan Deny, Pertandingan Ditolak dan Tidak ada kecocokan Tidak ditolak, jadi mungkin diizinkan oleh pernyataan lain.

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - webserver`	Pertandingan
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - webserver - test`	Pertandingan
`"ForAnyValue:StringEquals": {`	`aws:TagKeys: - environment - test`	Tidak ada kecocokan

Kondisi Kebijakan	Permintaan Konteks	Hasil
`"aws:TagKeys": "webserver" }`		Mungkin diizinkan oleh pernyataan lain.
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	Tidak `aws:TagKeys` dalam konteks permintaan.	Tidak ada kecocokan Mungkin diizinkan oleh pernyataan lain.

Single-valued contoh kebijakan utama konteks

Kumpulan contoh kebijakan berikut menunjukkan cara membuat kondisi kebijakan dengan kunci konteks bernilai tunggal.

Contoh: Beberapa blok kondisi dengan kunci konteks bernilai tunggal

Ketika blok kondisi memiliki beberapa kondisi, masing-masing dengan kunci konteks tunggal, semua kunci konteks harus diselesaikan menjadi true agar `Deny` efek yang diinginkan `Allow` atau dipanggil. Bila Anda menggunakan operator kondisi pencocokan yang dinegasikan, logika evaluasi nilai kondisi dibalik.

Contoh berikut memungkinkan pengguna membuat volume EC2 dan menerapkan tag ke volume selama pembuatan volume. Konteks permintaan harus menyertakan nilai untuk kunci konteks`aws:RequestTag/project`, dan nilai untuk kunci konteks `aws:ResourceTag/environment` dapat berupa apa saja kecuali produksi.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

Konteks permintaan harus menyertakan nilai tag proyek dan tidak dapat dibuat untuk sumber daya produksi untuk memanggil efek. `Allow` Volume EC2 berikut berhasil dibuat karena nama proyek `Feature3` dengan tag `QA` resource.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Contoh: Satu blok kondisi dengan beberapa kunci konteks bernilai tunggal dan nilai

Ketika blok kondisi berisi beberapa kunci konteks dan setiap kunci konteks memiliki beberapa nilai, setiap kunci konteks harus diselesaikan menjadi true untuk setidaknya satu nilai kunci agar `Deny` efek yang diinginkan `Allow` atau dipanggil. Saat Anda menggunakan operator kondisi pencocokan yang dinegasikan, logika evaluasi nilai kunci konteks dibalik.

Contoh berikut memungkinkan pengguna untuk memulai dan menjalankan tugas di cluster Amazon Elastic Container Service.

- Konteks permintaan harus menyer `production` takan OR `prod-backup` untuk kunci `aws:RequestTag/environment` konteks DAN.
- K `ecs:cluster` unci konteks memastikan bahwa tugas dijalankan pada cluster `default1` OR `default2` ARN ECS.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

Tabel berikut menunjukkan cara AWS mengevaluasi kebijakan ini berdasarkan nilai kunci kondisi dalam permintaan Anda.

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default1`	Cocokkan
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:prod-backup ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Cocokkan

Kondisi kebijakan	Permintaan konteks	Hasil
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: webserver:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Tidak cocok
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	Tidak `aws:RequestTag` dalam konteks permintaan. `ecs:cluster arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Tidak cocok

Elemen kebijakan IAM: Variabel dan tag

Gunakan variabel kebijakan AWS Identity and Access Management (IAM) sebagai placeholder jika Anda tidak mengetahui nilai pasti sumber daya atau kunci kondisi saat menulis kebijakan.

> **Note**
>
> Jika AWS tidak dapat menyelesaikan variabel, ini dapat menyebabkan seluruh pernyataan menjadi tidak valid. Misalnya, jika Anda menggunakan variabel `aws:TokenIssueTime`, variabel tersebut hanya menghasilkan nilai saat pemohon diautentikasi menggunakan kredensial sementara (peran IAM). Untuk mencegah variabel menyebabkan pernyataan tidak valid, gunakan... IfExists kondisi operator.

Topik

- Pengantar
- Menggunakan variabel dalam kebijakan
- Tanda sebagai variabel kebijakan
- Tempat Anda dapat menggunakan variabel kebijakan
- Variabel kebijakan tanpa nilai
- Meminta informasi yang dapat Anda gunakan untuk variabel kebijakan
- Menentukan nilai default
- Untuk informasi selengkapnya

Pengantar

Dalam kebijakan IAM, banyak tindakan memungkinkan Anda untuk memberikan nama untuk sumber daya spesifik yang Anda ingin kontrol aksesnya. Misalnya, kebijakan berikut memungkinkan pengguna untuk membuat daftar, membaca, dan menulis objek di bucket S3 `amzn-s3-demo-bucket` untuk `marketing` proyek.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/marketing/*"]
    }
  ]
}
```

Dalam beberapa kasus, Anda mungkin tidak mengetahui nama pasti sumber daya saat Anda menyusun kebijakan. Anda mungkin ingin melakukan generalisasi kebijakan sehingga dapat digunakan oleh banyak pengguna tanpa harus membuat salinan unik dari kebijakan tersebut untuk setiap pengguna. Alih-alih membuat kebijakan terpisah untuk setiap pengguna, sebaiknya buat kebijakan grup tunggal yang berfungsi untuk setiap pengguna dalam grup tersebut.

Menggunakan variabel dalam kebijakan

Anda dapat menentukan nilai dinamis di dalam kebijakan dengan menggunakan variabel kebijakan yang menetapkan placeholder dalam kebijakan.

Variabel ditandai menggunakan `$` awalan diikuti oleh sepasang kurung kur `{ }` ung () yang menyertakan nama variabel nilai dari permintaan.

Saat kebijakan dievaluasi, variabel kebijakan diganti dengan nilai yang berasal dari kunci konteks kondisional yang diteruskan dalam permintaan. Variabel dapat digunakan dalam kebijakan berbasis identitas, kebijakan sumber daya, kebijakan kontrol layanan, kebijakan sesi, dan kebijakan titik akhir VPC. Identity-based kebijakan yang digunakan sebagai batas izin juga mendukung variabel kebijakan.

Kunci konteks kondisi global dapat digunakan sebagai variabel dalam permintaan di seluruh AWS layanan. Kunci kondisi khusus layanan juga dapat digunakan sebagai variabel saat berinteraksi dengan AWS sumber daya, tetapi hanya tersedia ketika permintaan dibuat terhadap sumber daya yang mendukungnya. Untuk daftar kunci konteks yang tersedia untuk setiap AWS layanan dan

sumber daya, lihat Referensi O torisasi Layanan . Dalam keadaan tertentu, Anda tidak dapat mengisi kunci konteks kondisi global dengan nilai. Untuk mempelajari lebih lanjut tentang setiap kunci, lihat kunci konteks kondisi AWS global.

> **Important**
>
> - Nama utama peka dengan huruf besar dan kecil. Misalnya, `aws:CurrentTime` setara dengan `AWS:currenttime`.
> - Anda dapat menggunakan kunci kondisi bernilai tunggal sebagai variabel. Anda tidak dapat menggunakan kunci kondisi multivalue sebagai variabel.

Contoh berikut menunjukkan kebijakan untuk peran atau pengguna IAM yang menggantikan nama sumber daya tertentu dengan variabel kebijakan. Anda dapat menggunakan kembali kebijakan ini dengan memanfaatkan kunci `aws:PrincipalTag` kondisi. Saat kebijakan ini dievaluasi, `${aws:PrincipalTag/team}` tindakan hanya diperbolehkan jika nama bucket diakhiri dengan nama tim dari tag `team` utama.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

Variabel ditandai menggunakan prefiks $ yang diikuti dengan sepasang rangka yang melengkung ({ }). Di dalam ${ } karakter, Anda dapat menyertakan nama nilai dari permintaan yang ingin Anda gunakan dalam kebijakan. Nilai-nilai yang dapat Anda gunakan akan dibahas nanti di halaman ini.

Untuk detail tentang kunci kondisi global ini, lihat aw:PrincipalTag/kunci tag dalam daftar kunci kondisi global.

> **Note**
>
> Untuk menggunakan variabel kebijakan, Anda harus menyertakan elemen `Version` dalam pernyataan, dan versinya harus diatur ke versi yang mendukung variabel kebijakan. Variabel diperkenalkan dalam versi `2012-10-17`. Versi sebelumnya dari bahasa kebijakan tidak mendukung variabel kebijakan. Jika Anda tidak menyertakan elemen `Version` dan mengaturnya ke tanggal versi yang sesuai, variabel seperti `${aws:username}` diperlakukan sebagai string literal dalam kebijakan ini.
> Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Sebaliknya, versi kebijakan dibuat saat Anda mengubah kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola. Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat the section called “Version”. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called “Peningkatan versi IAM”.

Kebijakan yang memungkinkan kepala sekolah untuk mendapatkan objek dari jalur /David dari bucket S3 terlihat seperti ini:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
```

```
        "arn:aws:s3:::amzn-s3-demo-bucket/David/*"
      ]
    }
  ]
}
```

Jika kebijakan ini dilampirkan ke penggunaDavid, pengguna tersebut mendapatkan objek dari bucket S3-nya sendiri, tetapi Anda harus membuat kebijakan terpisah untuk setiap pengguna yang menyertakan nama pengguna. Kemudian, Anda akan menerapkan setiap kebijakan kepada pengguna individu.

Dengan menggunakan variabel kebijakan, Anda dapat membuat kebijakan yang dapat digunakan kembali. Kebijakan berikut memungkinkan pengguna mendapatkan objek dari bucket Amazon S3 jika nilai tag-key untuk `aws:PrincipalTag` cocok dengan nilai tag-key yang diteruskan dalam `owner` permintaan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Bila Anda menggunakan variabel kebijakan sebagai pengganti pengguna seperti ini, Anda tidak harus memiliki kebijakan terpisah untuk setiap pengguna individu. Dalam contoh berikut, kebijakan dilampirkan ke peran IAM yang diasumsikan oleh Manajer Produk menggunakan kredentif keamanan sementara. Saat pengguna membuat permintaan untuk menambahkan objek Amazon S3, IAM

mengganti nilai `dept` tag dari permintaan saat ini untuk `${aws:PrincipalTag}` variabel dan mengevaluasi kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOnlyDeptS3Prefix",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/dept}/*"
            ]
        }
    ]
}
```

Tanda sebagai variabel kebijakan

Di beberapa AWS layanan, Anda dapat melampirkan atribut khusus Anda sendiri ke sumber daya yang dibuat oleh layanan tersebut. Misalnya, Anda dapat menerapkan tag ke bucket Amazon S3 atau pengguna IAM. Tanda ini adalah pasangan nilai kunci. Anda menentukan nama kunci tanda dan nilai yang terkait dengan nama kunci tersebut. Misalnya, Anda dapat membuat tanda dengan kunci **`department`** dan nilai **`Human Resources`**. Untuk informasi lebih lanjut tentang pemberian tag pada entitas IAM, lihat Tag untuk AWS Identity and Access Management sumber daya. Untuk informasi tentang penandaan sumber daya yang dibuat oleh layanan AWS lainnya, lihat dokumentasi untuk layanan tersebut. Untuk informasi tentang penggunaan Tag Editor, lihat Bekerja dengan Tag Editor dalam Panduan Pengguna Konsol Manajemen AWS .

Anda dapat menandai sumber daya IAM untuk menyederhanakan menemukan, mengatur, dan melacak sumber daya IAM Anda. Anda juga dapat menandai identitas IAM untuk mengontrol akses ke sumber daya atau menandainya sendiri. Untuk mempelajari lebih lanjut tentang penggunaan tag untuk mengontrol akses, lihat Mengontrol akses ke dan untuk pengguna dan peran IAM menggunakan tag.

Tempat Anda dapat menggunakan variabel kebijakan

Anda dapat menggunakan variabel kebijakan di elemen `Resource` dan dalam perbandingan string dalam elemen `Condition`.

Elemen sumber daya

Anda dapat menggunakan variabel kebijakan di elemen `Resource`, tetapi hanya di bagian sumber daya ARN. Bagian ARN ini muncul setelah tanda titik dua (:) yang kelima. Anda tidak dapat menggunakan variabel untuk mengganti bagian ARN sebelum tanda titik dua yang kelima, seperti layanan atau akun. Untuk informasi selengkapnya tentang format ARN, lihat ARN IAM.

Untuk mengganti bagian ARN dengan nilai tag, kelilingi awalan dan nama kunci dengan`${ }`. Misalnya, elemen Resource berikut hanya mengacu pada bucket yang diberi nama sama dengan nilai dalam tag departemen pengguna yang meminta.

```
"Resource": ["arn:aws::s3:::amzn-s3-demo-bucket/
${aws:PrincipalTag/department}"]
```

Banyak AWS sumber daya menggunakan ARN yang berisi nama yang dibuat pengguna. Kebijakan IAM berikut memastikan bahwa hanya pengguna yang dituju dengan nilai tag access-project, access-application, dan access-environment yang cocok yang dapat mengubah sumber dayanya. Selain itu, menggunakan * wildcard cocokkan, mereka dapat mengizinkan akhiran nama sumber daya khusus.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
```

```
}
```

Elemen kondisi

Anda dapat menggunakan variabel kebijakan untuk nilai `Condition` dalam setiap kondisi yang melibatkan operator string atau operator ARN. Operator string termasuk `StringEquals`, `StringLike`, dan `StringNotLike`. Operator ARN termasuk `ArnEquals` dan `ArnLike`. Anda tidak dapat menggunakan variabel kebijakan dengan operator lainnya, seperti operator `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address`, atau `Null`. Untuk informasi selengkapnya tentang operator kondisi, lihat Elemen kebijakan JSON IAM: Operator kondisi.

Saat mereferensikan tanda di dalam ekspresi elemen `Condition`, gunakan prefiks dan nama kunci yang relevan sebagai kunci kondisi. Kemudian gunakan nilai yang ingin Anda uji di dalam nilai kondisi.

Misalnya, contoh kebijakan berikut memungkinkan akses penuh ke pengguna, tetapi hanya jika tag `costCenter` dilampirkan ke pengguna. Tanda harus memiliki nilai `12345` atau `67890`. Jika tanda tidak memiliki nilai, atau memiliki nilai lain, permintaan akan gagal.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variabel kebijakan tanpa nilai

Ketika variabel kebijakan mereferensikan kunci konteks kondisi yang tidak memiliki nilai atau tidak ada dalam konteks otorisasi untuk permintaan, nilainya secara efektif nol. Tidak ada nilai yang sama atau sama. Kunci konteks kondisi mungkin tidak ada dalam konteks otorisasi ketika:

- Anda menggunakan kunci konteks kondisi khusus layanan dalam permintaan ke sumber daya yang tidak mendukung kunci kondisi tersebut.
- Tag pada prinsipal, sesi, sumber daya, atau permintaan IAM tidak ada.
- Keadaan lain seperti yang tercantum untuk setiap kunci konteks kondisi globalAWS kunci konteks kondisi global.

Bila Anda menggunakan variabel tanpa nilai dalam elemen kondisi kebijakan IAM, Elemen kebijakan JSON IAM: Operator kondisi suka `StringEquals` atau `StringLike` tidak cocok, dan pernyataan kebijakan tidak berlaku.

Operator kondisi terbalik menyukai `StringNotEquals` atau `StringNotLike` cocok dengan nilai nol, karena nilai kunci kondisi yang mereka uji tidak sama dengan atau seperti nilai nol yang efektif.

Dalam contoh berikut, `aws:principaltag/Team` harus sama dengan mengi `s3:ExistingObjectTag/Team` zinkan akses. Akses secara eksplisit ditolak ketika `aws:principaltag/Team` tidak disetel. Jika variabel yang tidak memiliki nilai dalam konteks otorisasi digunakan sebagai bagian dari `NotResource` elemen `Resource` atau kebijakan, sumber daya yang menyertakan variabel kebijakan tanpa nilai tidak akan cocok dengan sumber daya apa pun.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
```

```
      }
    }
  ]
}
```

Meminta informasi yang dapat Anda gunakan untuk variabel kebijakan

Anda dapat menggunakan elemen `Condition` dari kebijakan JSON untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda. Saat Anda menggunakan variabel kebijakan, AWS ganti nilai dari kunci konteks permintaan sebagai pengganti variabel dalam kebijakan Anda.

Nilai-nilai kunci utama

Nilai untuk `aws:username`, `aws:userid`, dan `aws:PrincipalType` bergantung pada tipe prinsipal yang memulai permintaan. Misalnya, permintaan dapat dibuat menggunakan kredenSIAL pengguna IAM, peran IAM, atau. Pengguna root akun AWS tabel berikut menunjukkan nilai-nilai untuk kunci ini untuk berbagai tipe prinsipal.

- Pengguna root akun AWS
  - `aws:username`: (tidak ada)
  - `aws:userid`: Akun AWS ID
  - `aws:PrincipalType`: `Account`
- Pengguna IAM
  - `aws:username`: *`IAM-user-name`*
  - `aws:userid`: ID unik
  - `aws:PrincipalType`: `User`
- AWS STS prinsipal pengguna federasi
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Prinsipal federasi OIDC dan SAML

> **Note**
>
> Untuk informasi tentang kunci kebijakan yang tersedia saat Anda menggunakan federasi OIDC, lihatFederasi OIDC.

  - `aws:username`: (tidak ada)
  - `aws:userid`: (tidak ada)
  - `aws:PrincipalType`: `AssumedRole`
- Peran yang diasumsikan
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`role-id`*`:`*`caller-specified-role-name`*
  - `aws:PrincipalType`: `Assumed role`
- Peran yang ditetapkan ke instans Amazon EC2
  - `aws:username`: (tidak ada)
  - `aws:userid`: *`role-id`*`:`*`ec2-instance-id`*
  - `aws:PrincipalType`: `Assumed role`
- Penelepon anonim (khusus Amazon SQS, Amazon SNS, dan Amazon S3)
  - `aws:username`: (tidak ada)
  - `aws:userid`: (tidak ada)
  - `aws:PrincipalType`: `Anonymous`

Untuk item dalam daftar ini, ingat hal berikut:

- tidak ada berarti bahwa nilai tersebut tidak tercantum dalam informasi permintaan saat ini, dan setiap upaya untuk mencocokannya gagal dan menyebabkan pernyataan tersebut menjadi tidak valid.
- *`role-id`*adalah pengenal unik yang ditetapkan untuk setiap peran saat pembuatan. Anda dapat menampilkan ID peran dengan AWS CLI perintah: `aws iam get-role --role-name` *`rolename`*

- *caller-specified-name*dan *caller-specified-role-name* adalah nama yang dilewatkan oleh proses panggilan (seperti aplikasi atau layanan) ketika melakukan panggilan untuk mendapatkan kredenSIAL sementara.
- *ec2-instance-id*adalah nilai yang ditetapkan untuk instance saat diluncurkan dan muncul di halaman Instans konsol Amazon EC2. Anda juga dapat menampilkan ID instance dengan menjalankan AWS CLI perintah: `aws ec2 describe-instances`

Informasi tersedia dalam permintaan untuk kepala sekolah federasi

Prinsipal federasi adalah pengguna yang diautentikasi menggunakan sistem selain IAM. Misalnya, perusahaan mungkin memiliki aplikasi untuk digunakan di rumah yang melakukan panggilan ke AWS. Mungkin tidak praktis untuk memberikan identitas IAM kepada setiap pengguna perusahaan yang menggunakan aplikasi. Sebaliknya, perusahaan mungkin menggunakan aplikasi proksi (tingkat menengah) yang memiliki satu identitas IAM, atau perusahaan mungkin menggunakan penyedia identitas (IdP) SAML. Aplikasi proksi atau SAML IdP mengautentikasi pengguna individu yang menggunakan jaringan perusahaan. Aplikasi proksi kemudian dapat menggunakan identitas IAM-nya untuk mendapatkan kredensial keamanan sementara bagi pengguna individu. IdP SAML dapat secara efektif bertukar informasi identitas untuk kredenSIAL keamanan AWS sementara. KredenSIAL sementara kemudian dapat digunakan untuk mengakses AWS sumber daya.

Demikian pula, Anda dapat membuat aplikasi untuk perangkat seluler dengan aplikasi yang perlu mengakses sumber daya AWS . Dalam hal ini, Anda dapat menggunakan federasi OIDC, di mana aplikasi mengotentikasi pengguna menggunakan penyedia identitas terkenal seperti Login with Amazon, Amazon Cognito, Facebook, atau Google. Kemudian aplikasi dapat menggunakan informasi autentikasi pengguna dari penyedia ini untuk mendapatkan kredensial keamanan sementara guna mengakses sumber daya AWS .

Cara yang disarankan untuk menggunakan federasi OIDC adalah dengan memanfaatkan Amazon Cognito dan SDK AWS seluler. Untuk informasi selengkapnya, lihat berikut ini:

- Panduan Pengguna Amazon Cognito
- Skenario umum untuk kredensial sementara

Karakter khusus

Ada beberapa variabel kebijakan khusus yang telah ditetapkan sebelumnya yang memiliki nilai tetap yang memungkinkan Anda mewakili karakter yang memiliki arti khusus. Jika karakter khusus

ini adalah bagian dari string, Anda mencoba untuk mencocokkannya dan Anda memasukkannya secara literal sehingga mereka akan disalahartikan. Misalnya, menyisipkan tanda bintang * dalam string akan ditafsirkan sebagai wildcard yang cocok dengan karakter apa pun, bukan sebagai tanda bintang * yang sebenarnya. Dalam kasus ini, Anda dapat menggunakan variabel kebijakan yang telah ditentukan sebelumnya berikut ini:

- ${*} - gunakan jika Anda memerlukan karakter * (tanda bintang).
- ${?} - gunakan jika Anda memerlukan katakter ? (tanda tanya).
- ${$} - gunakan jika Anda memerlukan karakter $ (simbol dolar).

Variabel kebijakan yang sudah ditentukan sebelumnya ini dapat digunakan dalam string apa pun tempat Anda dapat menggunakan variabel kebijakan reguler.

Menentukan nilai default

Untuk menambahkan nilai default ke variabel, sertai nilai default dengan tanda kutip tunggal (`' '`), dan pisahkan teks variabel serta nilai default dengan koma dan spasi (`, `).

Misalnya, jika kepala sekolah diberi tag`team=yellow`, mereka dapat mengakses bucket `ExampleCorp's` Amazon S3 bernama`amzn-s3-demo-bucket-yellow`. Kebijakan dengan sumber daya ini memungkinkan anggota tim mengakses bucket tim mereka, tetapi tidak untuk tim lain. Untuk pengguna tanpa tag tim, ini menetapkan nilai default `company-wide` untuk nama bucket. Pengguna ini hanya dapat mengakses `amzn-s3-demo-bucket-company-wide` bucket tempat mereka dapat melihat informasi yang luas, seperti instruksi untuk bergabung dengan tim.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Untuk informasi selengkapnya

Untuk informasi selengkapnya tentang kebijakan, lihat hal berikut:

- Kebijakan dan izin di AWS Identity and Access Management
- Contoh kebijakan berbasis identitas IAM
- Referensi elemen kebijakan IAM JSON
- Logika evaluasi kebijakan
- Federasi OIDC

Elemen kebijakan JSON IAM: Jenis data yang didukung

Bagian ini mencantumkan tipe data yang didukung ketika Anda menentukan nilai dalam kebijakan JSON. Bahasa kebijakan tidak mendukung semua tipe untuk setiap elemen kebijakan; untuk informasi tentang setiap elemen, lihat bagian sebelumnya.

- String
- Bilangan (Bulat dan Desimal)
- Boolean
- Nol
- Daftar
- Peta
- Struktur (yang hanya berupa Peta bersarang)

Tabel berikut memetakan setiap tipe data ke serialisasi. Perhatikan bahwa semua kebijakan harus ada UTF-8. Untuk informasi tentang tipe data JSON, lihat RFC 46 27.

Tipe	JSON
String	String
Bulat	Bilangan
Desimal	Bilangan
Boolean	salah benar
Nol	nol
Tanggal	String sesuai dengan Profil W3C ISO 8601
IpAddress	String sesuai dengan RFC 4632
Daftar	Susunan
Objek	Objek

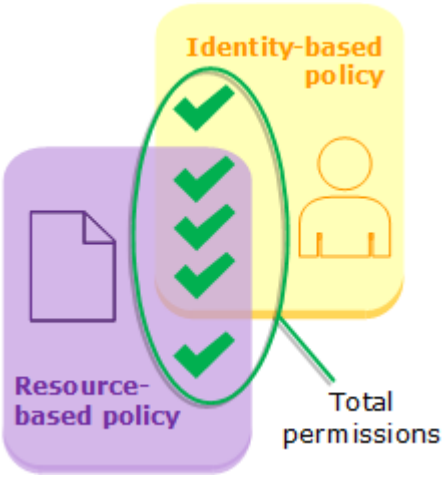
Logika evaluasi kebijakan

Ketika kepala sekolah mencoba menggunakan Konsol Manajemen AWS, AWS API, atau, prinsipal tersebut AWS CLI mengirimkan permintaan ke AWS. Ketika AWS layanan menerima permintaan, AWS selesaikan beberapa langkah untuk menentukan apakah akan mengizinkan atau menolak permintaan tersebut.

1. Otentikasi — AWS pertama mengotentikasi prinsipal yang membuat permintaan, jika perlu. Langkah ini tidak diperlukan untuk beberapa layanan, seperti Amazon S3 yang memungkinkan beberapa permintaan dari pengguna anonim.
2. Memproses konteks permintaan AWS Memproses informasi yang dikumpulkan dalam permintaan untuk menentukan kebijakan mana yang berlaku untuk permintaan tersebut.
3. Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses AWS Mengevaluasi semua jenis kebijakan dan urutan kebijakan mempengaruhi bagaimana mereka dievaluasi. AWS kemudian memproses kebijakan terhadap konteks permintaan untuk menentukan apakah permintaan diizinkan atau ditolak.

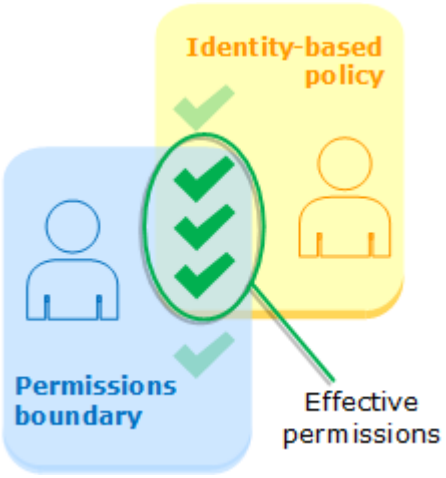
Mengevaluasi kebijakan berbasis identitas dengan kebijakan berbasis sumber daya

Identity-based kebijakan dan kebijakan berbasis sumber daya memberikan izin untuk identitas atau sumber daya yang dilampirkan. Ketika entitas IAM (pengguna atau peran) meminta akses ke sumber daya dalam akun yang sama, AWS mengevaluasi semua izin yang diberikan oleh kebijakan berbasis identitas dan sumber daya. Izin yang dihasilkan adalah gabungan dari izin dari kedua jenis. Jika suatu tindakan diizinkan oleh kebijakan berbasis identitas, kebijakan berbasis sumber daya, atau keduanya, maka izinkan tindakan tersebut. AWS Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.



Mengevaluasi kebijakan berbasis identitas dengan batas izin

Saat AWS mengevaluasi kebijakan berbasis identitas dan batas izin untuk pengguna, izin yang dihasilkan adalah persimpangan dari dua kategori. Artinya ketika Anda menambahkan batas izin ke pengguna dengan kebijakan berbasis identitas yang ada, Anda mungkin mengurangi tindakan yang dapat dilakukan pengguna. Atau, saat Anda menghapus batas izin dari pengguna, Anda mungkin meningkatkan tindakan yang dapat mereka lakukan. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin. Untuk melihat informasi tentang cara tipe kebijakan lain dievaluasi dengan batas izin, lihat Mengevaluasi izin efektif dengan batasan.



Mengevaluasi kebijakan berbasis identitas dengan AWS Organizations SCP atau RCP

Bila pengguna termasuk dalam akun yang merupakan anggota organisasi dan mengakses sumber daya yang tidak memiliki kebijakan berbasis sumber daya yang dikonfigurasi, izin yang dihasilkan adalah persimpangan kebijakan pengguna, kebijakan kontrol layanan (SCP), dan kebijakan kontrol sumber daya (RCP). Ini berarti bahwa suatu tindakan harus diizinkan oleh ketiga jenis kebijakan. Penolakan eksplisit dalam kebijakan berbasis identitas, SCP, atau RCP mengesampingkan izin.



Anda dapat mempelajari apakah akun Anda adalah anggota organisasi di AWS Organizations. Anggota organisasi mungkin terpengaruh oleh SCP atau RCP. Untuk melihat data ini menggunakan AWS CLI perintah atau operasi AWS API, Anda harus memiliki izin untuk `organizations:DescribeOrganization` tindakan untuk AWS Organizations entitas Anda. Anda harus memiliki izin tambahan untuk melakukan operasi di AWS Organizations konsol. Untuk mengetahui apakah SCP atau RCP menolak akses ke permintaan tertentu, atau untuk mengubah izin efektif Anda, hubungi administrator Anda. AWS Organizations

Memproses konteks permintaan

Ketika AWS mengevaluasi dan mengotorisasi permintaan, itu mengumpulkan informasi permintaan ke dalam konteks permintaan. Konteks permintaan berisi informasi apa pun yang dapat digunakan dalam evaluasi kebijakan.

- Principal — Pengguna, peran, atau prinsipal pengguna federasi yang mengirim permintaan. Informasi tentang prinsipal mencakup kebijakan yang terkait dengan prinsipal tersebut
- Tindakan — Satu atau lebih tindakan yang ingin dilakukan kepala sekolah.
- Sumber Daya — Satu atau lebih objek AWS sumber daya di mana tindakan atau operasi dilakukan.
- Data sumber daya – Data terkait sumber daya yang diminta. Ini dapat mencakup informasi seperti nama tabel DynamoDB atau tag pada instans Amazon EC2.
- Data lingkungan – Informasi tentang alamat IP, agen pengguna, status yang diaktifkan SSL, atau waktu hari.

Informasi ini dibandingkan dengan kebijakan yang berlaku untuk menentukan apakah akan mengizinkan atau menolak permintaan. Anda dapat mengatur informasi properti ini menggunakan model Principal, Action, Resource, and Condition (PARC) untuk lebih memahami bagaimana AWS kebijakan dievaluasi.

Memahami model PARC

Model PARC mewakili konteks permintaan berdasarkan empat elemen JSON dalam bahasa kebijakan:

- PrincipalEntitas yang membuat permintaan. Prinsipal mewakili pengguna manusia atau beban kerja terprogram yang dapat diautentikasi dan kemudian diberi wewenang untuk melakukan tindakan di. Akun AWS

- ActionOperasi yang sedang dilakukan. Seringkali tindakan akan dipetakan ke tindakan API.
- Resource— AWS Sumber daya di mana tindakan sedang dilakukan.
- Condition— Kendala tambahan yang harus dipenuhi agar permintaan diizinkan.

Berikut ini menunjukkan contoh bagaimana model PARC dapat mewakili konteks permintaan:

```
Principal: AIDA123456789EXAMPLE
Action: s3:CreateBucket
Resource: arn:aws:s3:::amzn-s3-demo-bucket1
Context:
- aws:UserId=AIDA123456789EXAMPLE:BobsSession
- aws:PrincipalAccount=123456789012
- aws:PrincipalOrgId=o-example
- aws:PrincipalARN=arn:aws:iam::AIDA123456789EXAMPLE:role/HR
- aws:MultiFactorAuthPresent=true
- aws:CurrentTime=...
- aws:EpochTime=...
- aws:SourceIp=...
- aws:PrincipalTag/dept=123
- aws:PrincipalTag/project=blue
- aws:RequestTag/dept=123
```

Pentingnya konteks permintaan

Memahami konteks permintaan dan bagaimana berinteraksi dengan evaluasi kebijakan sangat penting untuk:

- Memecahkan masalah akses
- Merancang kebijakan yang efektif dan aman
- Memahami cakupan penuh izin yang diberikan oleh kebijakan
- Memprediksi hasil evaluasi kebijakan dalam skenario yang berbeda

Dengan memvisualisasikan konteks permintaan menggunakan model PARC, Anda dapat lebih mudah memahami cara AWS membuat keputusan otorisasi dan merancang kebijakan Anda secara lebih efektif.

Cara AWS menggunakan konteks permintaan

Saat mengevaluasi kebijakan, AWS membandingkan informasi dalam konteks permintaan dengan informasi yang ditentukan dalam semua kebijakan yang berlaku. Ini termasuk kebijakan berbasis identitas, kebijakan berbasis sumber daya, batas izin IAM, Organizations, Organizations, dan kebijakan sesi. SCPs RCPs

Untuk setiap jenis kebijakan, AWS gunakan konteks permintaan untuk memeriksa:

- Apakah kebijakan berlaku untuk permintaan berdasarkan prinsipal.
- Apakah tindakan yang diminta diizinkan pada sumber daya yang ditentukan.
- Apakah kondisi yang ditentukan dalam kebijakan dipenuhi oleh konteks permintaan.

Cara AWS mengevaluasi kebijakan tergantung pada jenis kebijakan yang berlaku untuk konteks permintaan. Jenis kebijakan ini tersedia untuk digunakan dalam satu Akun AWS. Untuk informasi selengkapnya tentang tipe kebijakan ini, lihat Kebijakan dan izin di AWS Identity and Access Management. Untuk mempelajari cara AWS mengevaluasi kebijakan untuk akses lintas akun, lihat. Cross-account logika evaluasi kebijakan

- AWS Organizations kebijakan kontrol sumber daya (RCPs) - AWS Organizations RCPs tentukan izin maksimum yang tersedia untuk sumber daya dalam akun di organisasi atau unit organisasi (OU). RCPs berlaku untuk sumber daya di akun anggota dan memengaruhi izin efektif untuk kepala sekolah, termasuk Pengguna root akun AWS, terlepas dari apakah prinsipal tersebut milik organisasi Anda. RCPs tidak berlaku untuk sumber daya di akun manajemen organisasi dan panggilan yang dilakukan oleh peran terkait layanan. Jika RCP hadir, izin yang diberikan oleh kebijakan berbasis identitas dan berbasis sumber daya ke sumber daya di akun anggota Anda hanya efektif jika RCP mengizinkan tindakan tersebut.
- AWS Organizations kebijakan kontrol layanan (SCPs) - AWS Organizations SCPs tentukan izin maksimum yang tersedia untuk prinsipal dalam akun di organisasi atau unit organisasi (OU). SCPs berlaku untuk kepala sekolah di akun anggota, termasuk masing-masing. Pengguna root akun AWS Jika SCP hadir, izin yang diberikan oleh kebijakan berbasis identitas dan berbasis sumber daya kepada kepala sekolah di akun anggota Anda hanya efektif jika SCP mengizinkan tindakan tersebut. Satu-satunya pengecualian adalah prinsip dalam akun manajemen organisasi dan peran terkait layanan.
- Kebijakan berbasis sumber daya — Kebijakan berbasis sumber daya memberikan izin untuk prinsipal yang ditentukan dalam kebijakan. Izin menentukan apa yang dapat dilakukan oleh prinsipal dengan sumber daya yang memiliki kebijakan tersebut.

- Batas izin — Batas izin adalah fitur yang menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna atau peran). Saat Anda menetapkan batas izin untuk entitas, entitas hanya dapat melakukan tindakan yang diizinkan oleh kebijakan berbasis identitas dan batas izinnya. Dalam beberapa kasus, penolakan implisit dalam batas izin dapat membatasi izin yang diberikan oleh kebijakan berbasis sumber daya. Untuk informasi selengkapnya, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.
- Kebijakan berbasis identitas – Kebijakan berbasis identitas diterapkan pada identitas IAM (pengguna, kelompok pengguna, atau peran) dan memberikan izin kepada entitas IAM (pengguna dan peran). Jika hanya kebijakan berbasis identitas yang berlaku untuk permintaan, maka AWS periksa semua kebijakan tersebut untuk setidaknya satu. `Allow`
- Kebijakan sesi — Kebijakan sesi adalah kebijakan yang Anda berikan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau sesi pengguna gabungan. Untuk membuat sesi peran secara terprogram, gunakan salah satu operasi API `AssumeRole*`. Saat Anda melakukan ini dan meneruskan kebijakan sesi, izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas entitas IAM dan kebijakan sesi. Untuk membuat sesi pengguna federasi, Anda menggunakan kunci akses pengguna IAM untuk memanggil operasi API secara terprogram. `GetFederationToken` Untuk informasi selengkapnya, lihat Kebijakan sesi.

Ingat, penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin.

> **Note**
>
> AWS Organizations Kebijakan deklaratif memungkinkan Anda untuk mendeklarasikan dan menerapkan konfigurasi yang Anda inginkan secara terpusat pada skala tertentu Layanan AWS di seluruh organisasi. Karena kebijakan deklaratif diterapkan secara langsung di tingkat layanan, kebijakan tersebut tidak berdampak langsung pada permintaan evaluasi kebijakan dan tidak disertakan dengan konteks permintaan. Untuk informasi selengkapnya, lihat Kebijakan deklaratif di Panduan AWS Organizations Pengguna.

Contoh evaluasi kebijakan menggunakan model PARC

Untuk mengilustrasikan bagaimana konteks permintaan berinteraksi dengan evaluasi kebijakan, mari pertimbangkan contoh kebijakan:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:CreateBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/dept": "123"
        }
      }
    }
  ]
}
```

Dalam contoh ini, kebijakan akan mengizinkan `CreateBucket` tindakan hanya jika konteks permintaan menyertakan `aws:PrincipalTag/dept` nilai “123" dan sumber daya cocok dengan nama `amzn-s3-demo-bucket1` bucket. Tabel berikut menunjukkan cara AWS menggunakan konteks permintaan untuk mengevaluasi kebijakan ini dan membuat keputusan otorisasi.

Kebijakan	Konteks permintaan	Hasil evaluasi

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3:Create Bucket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context: - aws:PrincipalTag/ dept=123`
		Pertandingan

`"StringEquals": { "aws:PrincipalTag/dept": "123"`	`Principal: AIDA12345 6789EXAMPLE`	Tidak cocok

Kebijakan	Konteks permintaan	Hasil evaluasi

`}`	`Action: s3: DeleteBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	`- aws:PrincipalTag/dept=123`	

`"StringEquals": {`
`"aws:PrincipalTag/dept": "123"`
`}`
	`Principal: AIDA123456789EXAMPLE`
	`Action: s3:CreateBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	`- aws:PrincipalTag/dept= 321`
		Tidak cocok

`"StringEquals": {`
`"aws:PrincipalTag/dept": "123"`
`}`
	`Principal: AIDA123456789EXAMPLE`
	`Action: s3:CreateBucket`
	`Resource: arn:aws:s3:::amzn-s3-demo-bucket1`
	`Context:`
	
	Tidak `aws:PrincipalTag` dalam permintaan.
		Tidak cocok

Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses

Kode AWS penegakan memutuskan apakah permintaan yang dikirim AWS harus diizinkan atau ditolak. AWS mengevaluasi semua kebijakan yang berlaku untuk konteks permintaan. Berikut ini adalah ringkasan dari logika evaluasi AWS kebijakan.

- Secara default, semua permintaan secara implisit ditolak dengan pengecualian Pengguna root akun AWS, yang memiliki akses penuh.
- Permintaan harus diizinkan secara eksplisit oleh kebijakan atau serangkaian kebijakan mengikuti logika evaluasi di bawah ini untuk diizinkan.
- Penolakan eksplisit mengesampingkan izin eksplisit.

Evaluasi kebijakan dapat berbeda tergantung pada apakah permintaan tersebut berada dalam satu akun atau permintaan lintas akun. Untuk detail tentang bagaimana keputusan evaluasi kebijakan dibuat untuk peran atau pengguna IAM dalam satu akun, lihatEvaluasi kebijakan untuk permintaan dalam satu akun. Untuk detail tentang bagaimana keputusan evaluasi kebijakan dibuat untuk permintaan lintas akun, lihatCross-account logika evaluasi kebijakan.

- Evaluasi penolakan – Secara default, semua permintaan ditolak. Ini disebut penolakan implisit. Kode AWS penegakan mengevaluasi semua kebijakan dalam akun yang berlaku untuk permintaan. Ini termasuk AWS Organizations SCP dan RCP, kebijakan berbasis sumber daya, kebijakan berbasis identitas, batas izin IAM, dan kebijakan sesi. Dalam semua kebijakan tersebut, kode penegakan mencari pernyataan `Deny` yang berlaku untuk permintaan. Ini disebut penolakan secara tegas. Jika kode penegakan menemukan bahkan satu penolakan eksplisit yang berlaku, kode penegakan mengembalikan keputusan akhir Den y. Jika tidak ada penolakan eksplisit, evaluasi kode penegakan berlanjut.
- AWS Organizations RCP — Kode penegakan mengevaluasi kebijakan kontrol AWS Organizations sumber daya (RCP) yang berlaku untuk permintaan. RCP berlaku untuk sumber daya akun tempat RCP dilampirkan. Jika kode penegakan tidak menemukan `Allow` pernyataan yang berlaku dalam RCP, kode penegakan mengembalikan keputusan akhir Deny. Perhatikan bahwa kebijakan ter AWS kelola yang `RCPFullAWSAccess` dipanggil secara otomatis dibuat dan dilampirkan ke setiap entitas di organisasi Anda termasuk root, setiap OU, dan Akun AWS saat RCP diaktifkan. `RCPFullAWSAccess`tidak dapat dilepaskan, jadi akan selalu ada `Allow` pernyataan. Jika tidak ada RCP, atau jika RCP mengizinkan tindakan yang diminta, evaluasi kode penegakan berlanjut.

- AWS Organizations SCPs — Kode penegakan mengevaluasi kebijakan kontrol AWS Organizations layanan (SCP) yang berlaku untuk permintaan. SCP berlaku untuk prinsipal akun tempat SCP dilampirkan. Jika kode penegakan tidak menemukan `Allow` pernyataan yang berlaku di SCP, kode penegakan mengembalikan keputusan akhir Deny. Jika tidak ada SCP, atau jika SCP mengizinkan tindakan yang diminta, evaluasi kode penegakan berlanjut.
- Resource-based Kebijakan — Dalam akun yang sama, kebijakan berbasis sumber daya berdampak pada evaluasi kebijakan secara berbeda tergantung pada jenis prinsipal yang mengakses sumber daya, dan prinsip yang diizinkan dalam kebijakan berbasis sumber daya. Bergantung pada jenis prinsipal, kebijakan berbasis sumber daya dapat menghasilkan keputusan akhir`Allow`, bahkan jika ada penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. `Allow`

  Untuk sebagian besar sumber daya, Anda hanya memerlukan keterangan eksplisit `Allow` untuk prinsipal baik dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya untuk memberikan akses. Kebijakan kepercayaan peran IAM dan kebijakan kunci KMS merupakan pengecualian untuk logika ini, karena kebijakan tersebut harus secara eksplisit mengizinkan akses untuk kepala sekolah. Resource-based kebijakan untuk layanan selain IAM dan AWS KMS mungkin juga memerlukan `Allow` pernyataan eksplisit dalam akun yang sama untuk memberikan akses. Untuk informasi selengkapnya, lihat dokumentasi untuk layanan tertentu yang Anda gunakan.

  Untuk permintaan evaluasi kebijakan akun tunggal, logika kebijakan berbasis sumber daya berbeda dari jenis kebijakan lainnya jika prinsipal yang ditentukan adalah pengguna IAM, peran IAM, atau prinsipal sesi. Prinsip sesi mencakup sesi peran IAM atau prinsipal pengguna AWS STS federasi. Jika kebijakan berbasis sumber daya memberikan izin langsung kepada pengguna IAM atau kepala sesi yang membuat permintaan, maka penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi tidak memengaruhi keputusan akhir.

  - Peran IAM — Resource-based kebijakan yang memberikan izin ke peran IAM ARN dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi. Anda dapat menentukan peran ARN dalam elemen Principal atau kunci `aws:PrincipalArn` kondisi. Dalam kedua kasus tersebut, prinsipal yang membuat permintaan adalah sesi peran IAM.

    Batas izin dan kebijakan sesi tidak membatasi izin yang diberikan menggunakan kunci `aws:PrincipalArn` kondisi dengan wildcard (*) di elemen Principal, kecuali kebijakan berbasis identitas berisi penolakan eksplisit. Untuk informasi selengkapnya, lihat Prinsip peran IAM.

    Contoh peran ARN

```
arn:aws:iam::111122223333:role/examplerole
```

- Sesi peran IAM — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin ke sesi peran IAM ARN memberikan izin langsung ke sesi peran yang diasumsikan. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi. Ketika Anda mengambil peran dan membuat permintaan, prinsipal yang membuat permintaan adalah sesi peran IAM ARN dan bukan ARN dari peran itu sendiri. Untuk informasi selengkapnya, lihat Prinsip sesi peran.

  Contoh sesi peran ARN

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- Pengguna IAM — Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna IAM ARN (yang bukan sesi pengguna federasi) tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas atau batas izin.

  Contoh pengguna IAM ARN

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- AWS STS Principal pengguna federasi — Sesi pengguna federasi adalah sesi yang dibuat dengan panggilan. `GetFederationToken` Ketika pengguna federasi membuat permintaan, prinsipal yang membuat permintaan adalah pengguna federasi ARN dan bukan ARN dari pengguna IAM yang berfederasi. Dalam akun yang sama, kebijakan berbasis sumber daya yang memberikan izin kepada pengguna federasi ARN memberikan izin langsung ke sesi. Izin yang diberikan langsung ke sesi tidak dibatasi oleh penolakan implisit dalam kebijakan berbasis identitas, batas izin, atau kebijakan sesi.

  Namun, jika kebijakan berbasis sumber daya memberikan izin kepada ARN pengguna IAM yang melakukan federasi, permintaan yang dibuat oleh pengguna federasi selama sesi dibatasi oleh penolakan implisit dalam batas izin atau kebijakan sesi.

  Contoh sesi pengguna federasi ARN

  ```
  arn:aws:sts::111122223333:federated-user/exampleuser
  ```

- Identity-based kebijakan — Kode penegakan memeriksa kebijakan berbasis identitas untuk prinsipal. Untuk pengguna IAM, ini termasuk kebijakan pengguna dan kebijakan dari grup tempat

pengguna berada. Jika tidak ada kebijakan berbasis identitas atau tidak ada pernyataan dalam kebijakan berbasis identitas yang mengizinkan tindakan yang diminta, maka permintaan ditolak secara implisit dan kode penegakan mengembalikan keputusan akhir Deny. Jika ada pernyataan dalam kebijakan berbasis identitas yang berlaku memungkinkan tindakan yang diminta, evaluasi kode berlanjut.

- Batas izin IAM — Kode penegakan memeriksa apakah entitas IAM yang digunakan oleh kepala sekolah memiliki batas izin. Jika kebijakan yang digunakan untuk menetapkan batas izin tidak mengizinkan tindakan yang diminta, maka permintaan tersebut ditolak secara implisit. Kode penegakan mengembalikan keputusan akhir Den y. Jika tidak ada batas izin, atau jika batas izin memungkinkan tindakan yang diminta, evaluasi kode berlanjut.
- Kebijakan sesi — Kode penegakan memeriksa apakah prinsipal adalah prinsipal sesi. Prinsip sesi mencakup sesi peran IAM atau sesi pengguna AWS STS federasi. Jika prinsipal bukan pokok sesi, kode penegakan mengembalikan keputusan akhir Iz inkan.

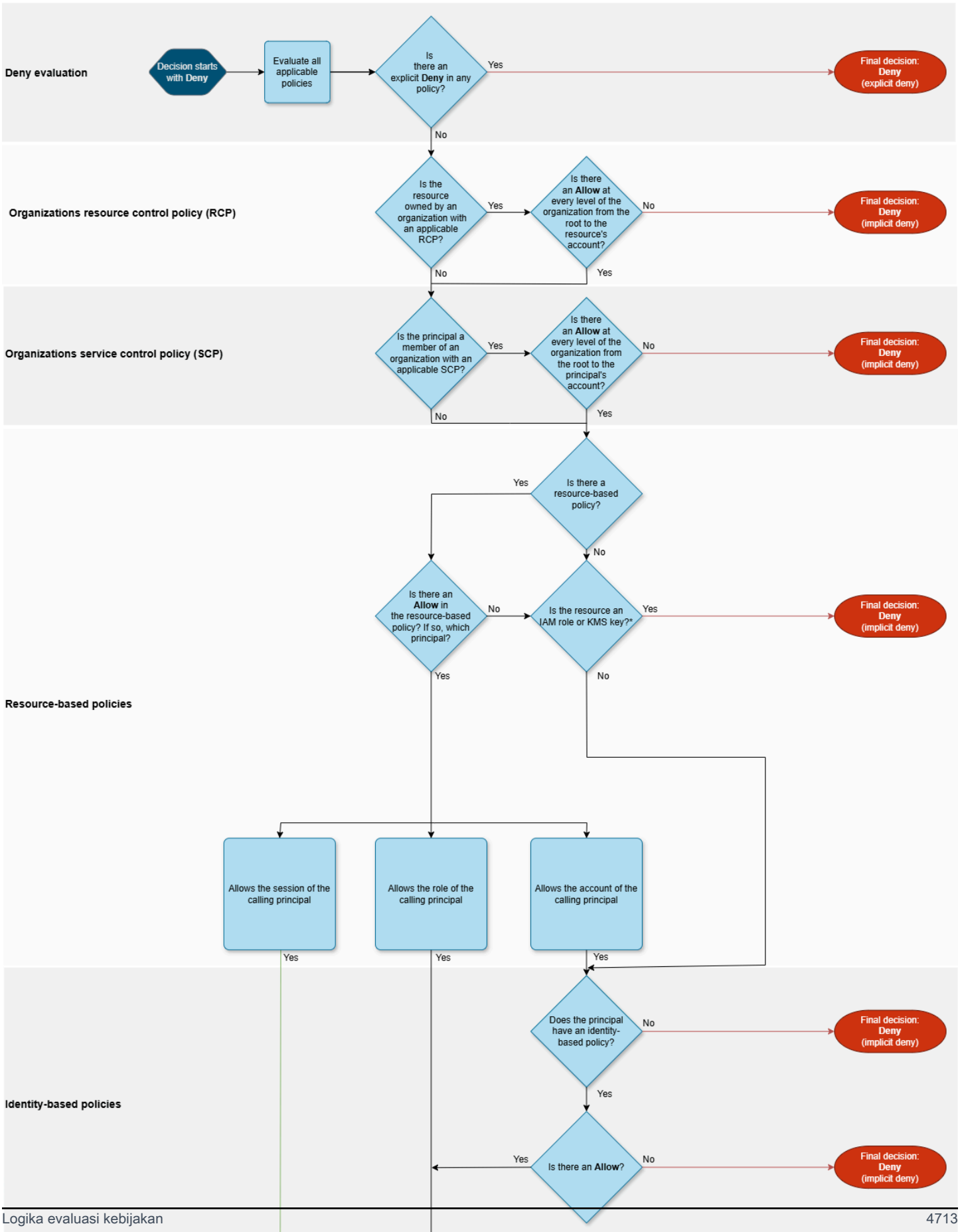
  Untuk prinsipal sesi, kode penegakan memeriksa apakah kebijakan sesi diteruskan dalam permintaan. Anda dapat meneruskan kebijakan sesi saat menggunakan AWS API AWS CLI atau untuk mendapatkan kredentif sementara untuk peran atau prinsipal pengguna AWS STS federasi. Jika Anda tidak meneruskan kebijakan sesi, kebijakan sesi default dibuat dan kode penegakan mengembalikan keputusan akhir Iz inkan.

  - Jika kebijakan sesi hadir dan tidak mengizinkan tindakan yang diminta, maka permintaan ditolak secara implisit. Kode penegakan mengembalikan keputusan akhir Den y.
  - Kode penegakan memeriksa apakah prinsipal adalah sesi peran. Jika prinsipal adalah sesi peran, maka permintaan tersebut Di izinkan. Jika tidak, permintaan ditolak secara implisit dan kode penegakan mengembalikan keputusan akhir Deny.
  - Jika ada kebijakan sesi dan mengizinkan tindakan yang diminta, maka kode penegakan mengembalikan keputusan akhir Iz inkan.

Evaluasi kebijakan untuk permintaan dalam satu akun

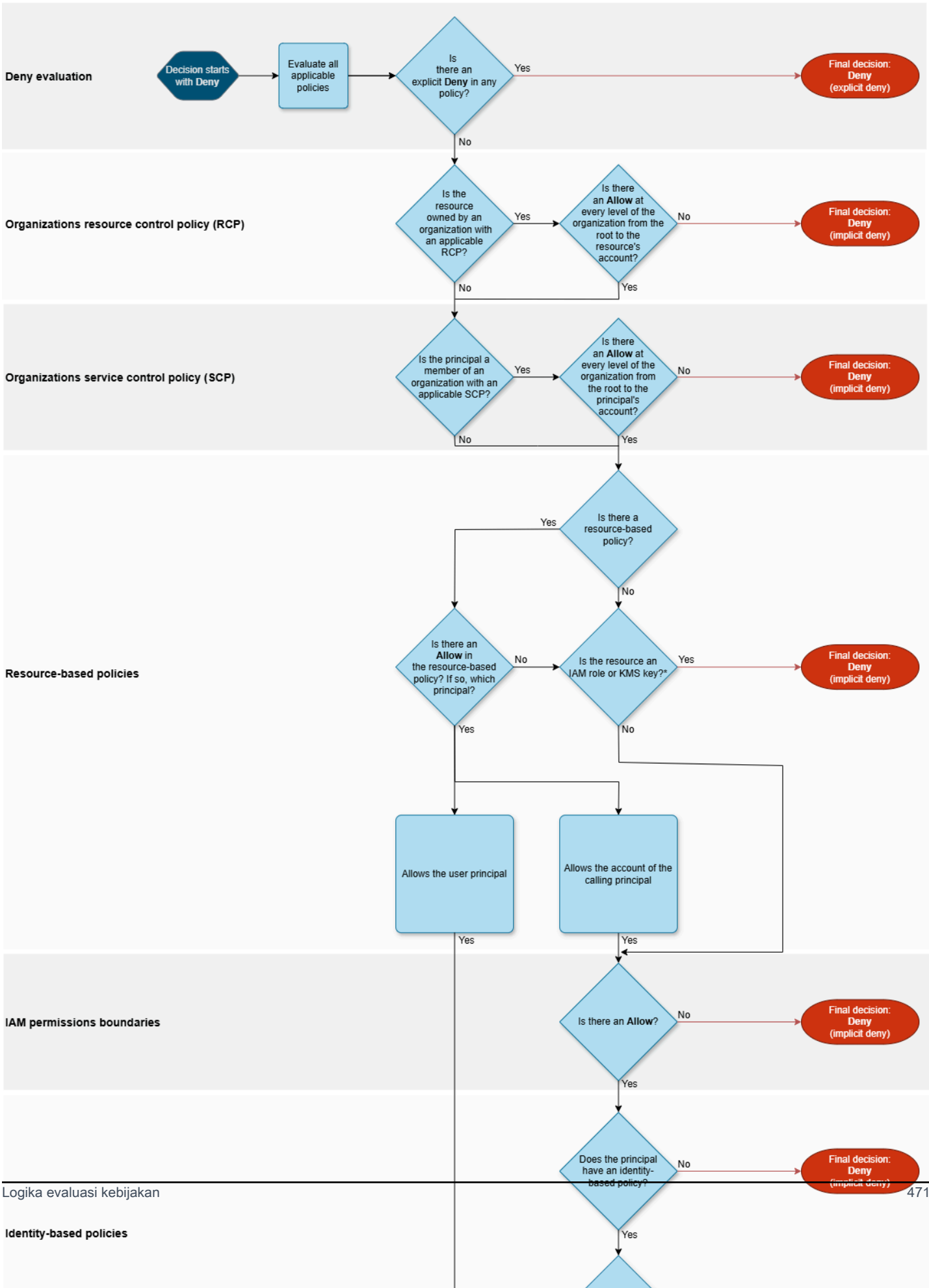
Evaluasi kebijakan untuk peran IAM

Diagram alir berikut memberikan rincian tentang bagaimana keputusan evaluasi kebijakan dibuat untuk peran IAM dalam satu akun.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Yes
Is there a resource-based policy?
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the session of the calling principal
Allows the role of the calling principal
Allows the account of the calling principal
Yes
Yes
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes
Yes
Is there an Allow?
No
Final decision: Deny (implicit deny)
IAM permissions boundaries
Does the principal have a permissions
Yes
Is there an Allow?
No
Final decision: Deny

Evaluasi kebijakan untuk pengguna IAM

Diagram alir berikut memberikan rincian tentang bagaimana keputusan evaluasi kebijakan dibuat untuk pengguna IAM dalam satu akun.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the user principal
Allows the account of the calling principal
Yes
IAM permissions boundaries
Is there an Allow?
No
Final decision: Deny (implicit deny)
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes

Contoh evaluasi kebijakan berbasis identitas dan kebijakan berbasis sumber daya

Tipe kebijakan yang paling umum adalah kebijakan berbasis identitas dan kebijakan berbasis sumber daya. Ketika akses ke sumber daya diminta, AWS evaluasi semua izin yang diberikan oleh kebijakan untuk setidaknya satu Izinkan dalam akun yang sama. Penolakan eksplisit dalam salah satu kebijakan mengesampingkan izin.

> **Important**
>
> Jika kebijakan berbasis identitas atau kebijakan berbasis sumber daya dalam akun yang sama mengizinkan permintaan dan yang lainnya tidak, permintaan tetap diizinkan.

Asumsikan bahwa Carlos memiliki nama pengguna `carlossalazar` dan dia mencoba menyimpan file ke bucket Amazon S3 `amzn-s3-demo-bucket-carlossalazar-logs`.

Juga asumsikan bahwa kebijakan berikut ini diberlakukan pada pengguna IAM `carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
```

```
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

Pernyataan `AllowS3ListRead` dalam kebijakan ini memungkinkan Carlos melihat daftar semua bucket di akun. Pernyataan `AllowS3Self` memungkinkan Carlos mendapatkan akses penuh ke bucket dengan nama yang sama dengan nama penggunanya. Pernyataan `DenyS3Logs` menolak akses Carlos ke setiap bucket S3 mana pun dengan `log` dalam namanya.

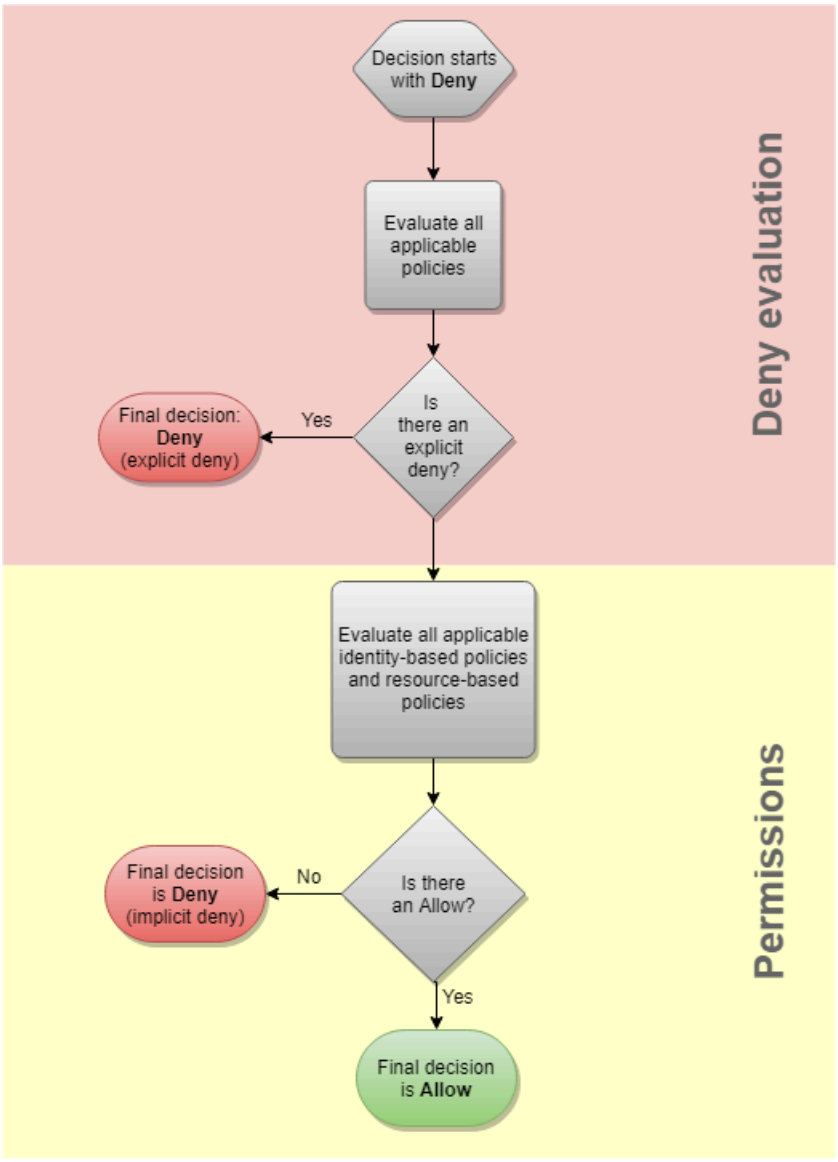
Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan buket) dilampirkan ke bucket `amzn-s3-demo-bucket-carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        }
    ]
}
```

Kebijakan ini menetapkan bahwa hanya pengguna `carlossalazar` yang dapat mengakses bucket `amzn-s3-demo-bucket-carlossalazar`.

Ketika Carlos membuat permintaannya untuk menyimpan file ke `amzn-s3-demo-bucket-carlossalazar-logs` ember, AWS tentukan kebijakan apa yang berlaku untuk permintaan tersebut. Dalam kasus ini, hanya kebijakan berbasis identitas dan kebijakan berbasis sumber daya yang berlaku. Keduanya adalah kebijakan izin. Karena batas izin tidak berlaku, logika evaluasi dikurangi ke logika berikut.



AWS pertama memeriksa `Deny` pernyataan yang berlaku untuk konteks permintaan. Ia menemukan satu, karena kebijakan berbasis identitas secara eksplisit menyangkal akses Carlos ke bucket S3 yang digunakan untuk logging. Akses Carlos ditolak.

Asumsikan bahwa dia kemudian menyadari kesalahannya dan mencoba menyimpan file ke `amzn-s3-demo-bucket-carlossalazar` ember. AWS memeriksa `Deny` pernyataan dan tidak menemukannya. Kemudian memeriksa kebijakan izin. Baik kebijakan berbasis identitas maupun kebijakan berbasis sumber daya mengizinkan permintaan tersebut. Karena itu, AWS memungkinkan permintaan. Jika salah satu dari mereka menolak pernyataan tersebut secara tegas, permintaan tersebut akan ditolak. Jika salah satu tipe kebijakan mengizinkan permintaan tersebut dan tipe yang lainnya tidak, permintaan tersebut masih diperbolehkan.

Cross-account logika evaluasi kebijakan

Anda dapat mengizinkan prinsipal di satu akun untuk mengakses sumber daya di akun kedua. Hal ini dikenal sebagai akses lintas akun. Saat Anda mengizinkan akses lintas akun, akun yang memiliki prinsipal disebut akun tepercaya. Akun yang memiliki sumber daya disebut akun kepercayaan.

Untuk memungkinkan akses lintas akun, Anda melampirkan kebijakan berbasis sumber daya ke sumber daya yang ingin Anda bagikan. Anda juga harus melampirkan kebijakan berbasis identitas pada identitas yang bertindak sebagai kepala sekolah dalam permintaan. Kebijakan berbasis sumber daya dalam akun kepercayaan harus menyebutkan prinsipal akun tepercaya yang akan memiliki akses ke sumber daya tersebut. Anda dapat menentukan seluruh akun atau pengguna IAM-nya, prinsip pengguna AWS STS federasi, peran IAM, atau sesi peran yang dianggap. Anda juga dapat menentukan AWS layanan sebagai kepala sekolah. Untuk informasi selengkapnya, lihat Cara menentukan prinsipal.

Kebijakan berbasis identitas prinsipal harus mengizinkan akses yang diminta ke sumber daya dalam layanan kepercayaan. Anda dapat melakukan ini dengan menentukan ARN sumber daya.

Di IAM, Anda dapat memberlakukan kebijakan berbasis sumber daya ke peran IAM agar prinsipal di akun lain dapat mengambil peran tersebut. Kebijakan berbasis sumber daya peran ini disebut kebijakan kepercayaan peran. Setelah mengambil peran tersebut, prinsipal yang diizinkan dapat menggunakan kredensial sementara yang dihasilkan untuk mengakses beberapa sumber daya dalam akun Anda. Akses ini ditetapkan dalam kebijakan izin berbasis identitas dari peran tersebut. Untuk mempelajari cara memungkinkan akses lintas akun menggunakan peran berbeda dengan mengizinkan akses lintas akun menggunakan kebijakan berbasis sumber daya lainnya, lihat Akses sumber daya lintas akun di IAM.

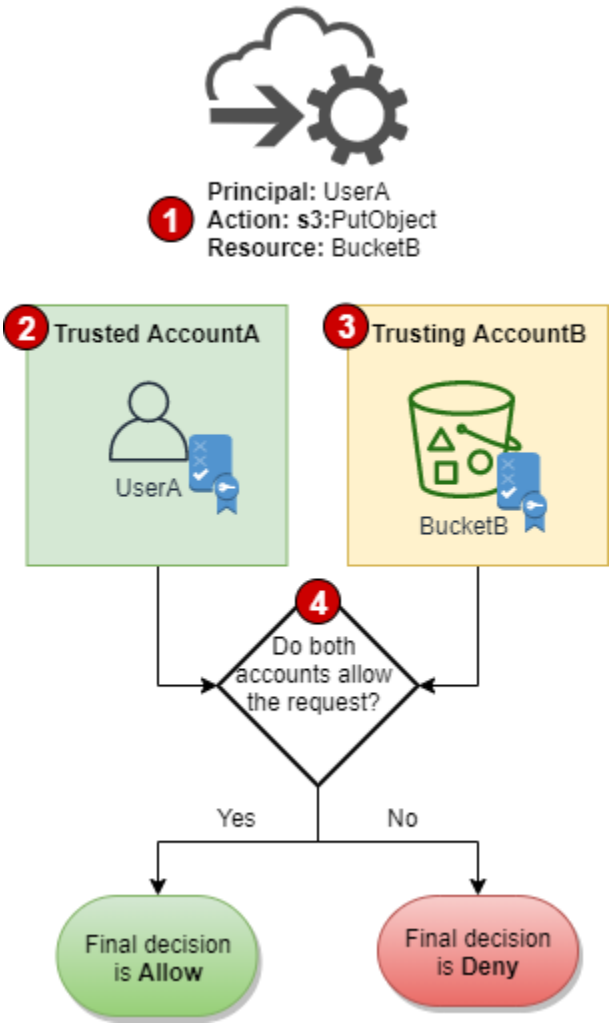
> **Important**
>
> Layanan lain dapat memengaruhi logika evaluasi kebijakan. Misalnya, AWS Organizations mendukung kebijakan kontrol layanan dan kebijakan kontrol sumber daya yang dapat

diterapkan pada prinsipal dan sumber daya dalam satu atau beberapa akun. AWS Resource Access Manager mendukung fragmen kebijakan yang mengontrol tindakan mana yang diizinkan dilakukan oleh prinsipal pada sumber daya yang dibagikan dengannya.

Menentukan apakah permintaan lintas akun diizinkan atau ditolak.

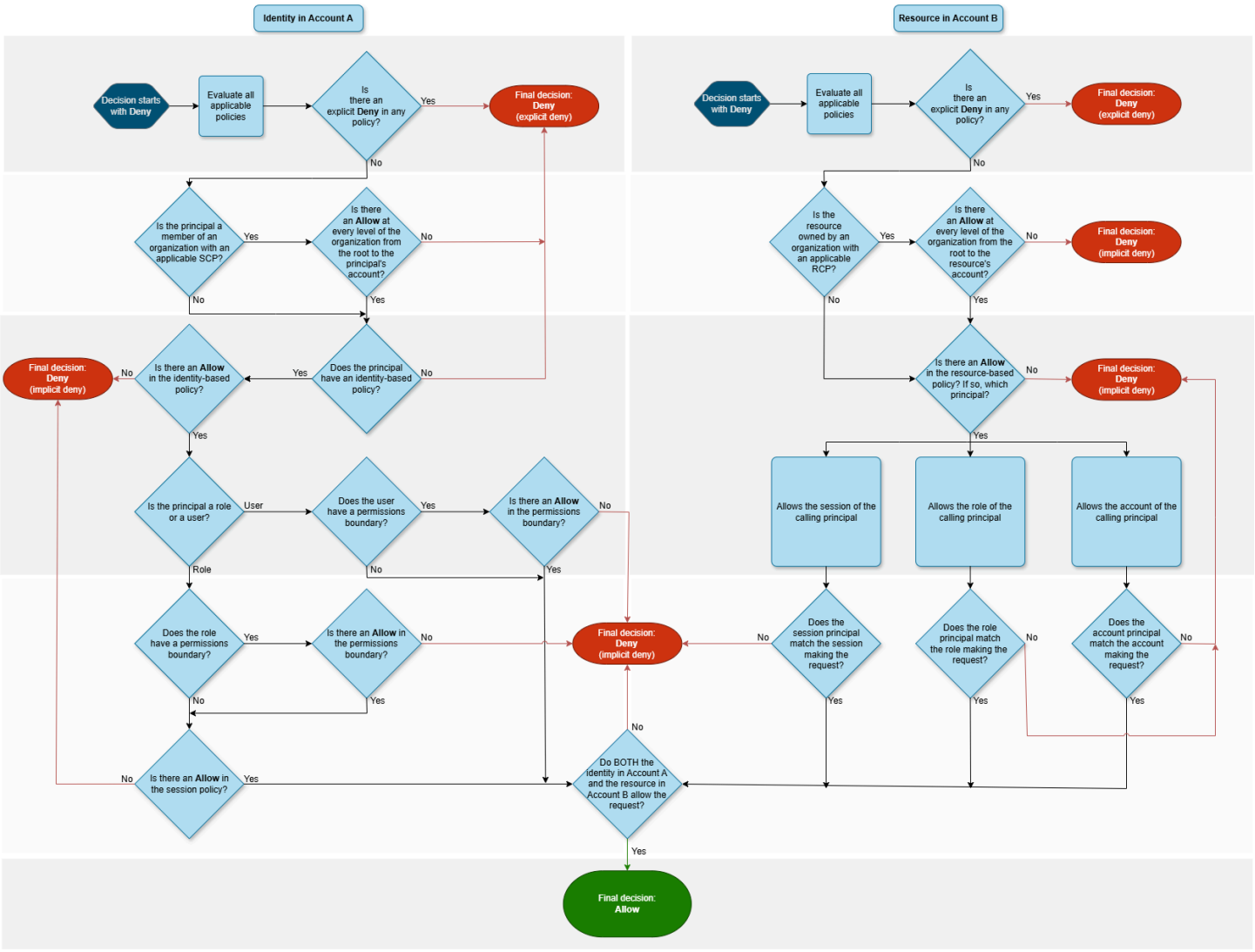
Untuk permintaan lintas akun, pemohon dalam `AccountA` tepercaya harus memiliki kebijakan berbasis identitas. Kebijakan yang harus memungkinkan mereka membuat permintaan ke sumber daya dalam `AccountB` kepercayaan. Selain itu, kebijakan berbasis sumber daya di `AccountB` harus mengizinkan pemohon di `AccountA` untuk mengakses sumber daya.

Saat Anda membuat permintaan lintas akun, AWS lakukan dua evaluasi. AWS mengevaluasi permintaan di akun kepercayaan dan akun tepercaya. Untuk informasi selengkapnya tentang bagaimana permintaan dievaluasi di satu akun, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses. Permintaan hanya diperbolehkan jika kedua evaluasi menghasilkan keputusan `Allow`.



1. Jika prinsipal di satu akun mengajukan permintaan untuk mengakses sumber daya di akun lain, ini adalah permintaan lintas akun.
2. Prinsipal mengajukan permohonan ada di akun tepercaya (`AccountA`). Saat AWS mengevaluasi akun ini, AWS memeriksa kebijakan berbasis identitas dan kebijakan apa pun yang dapat membatasi kebijakan berbasis identitas. Untuk informasi selengkapnya, lihat Mengevaluasi kebijakan berbasis identitas dengan batas izin.
3. Sumber daya yang diminta ada di akun kepercayaan (`AccountB`). Saat AWS mengevaluasi akun ini, AWS memeriksa kebijakan berbasis sumber daya yang dilampirkan ke sumber daya yang diminta dan setiap kebijakan yang dapat membatasi kebijakan berbasis sumber daya. Untuk informasi selengkapnya, lihat Mengevaluasi kebijakan berbasis identitas dengan kebijakan berbasis sumber daya.
4. AWS mengizinkan permintaan hanya jika kedua evaluasi kebijakan akun mengizinkan permintaan tersebut.

Diagram alir berikut memberikan ilustrasi yang lebih rinci tentang bagaimana keputusan evaluasi kebijakan dibuat untuk permintaan lintas akun. Sekali lagi, AWS izinkan permintaan hanya jika kedua evaluasi kebijakan akun mengizinkan permintaan tersebut.



Contoh evaluasi kebijakan lintas akun

Contoh berikut menunjukkan skenario di mana peran dalam satu akun diberikan izin oleh kebijakan berbasis sumber daya di akun kedua.

Asumsikan bahwa Carlos adalah pengembang dengan peran IAM yang disebutkan `Demo` di akun 11111111111111. Dia ingin menyimpan file ke bucket Amazon S3 `amzn-s3-demo-bucket-production-logs` di akun 222222222222.

Asumsikan juga bahwa kebijakan berikut ini melekat pada peran `Demo` IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

Pernyataan `AllowS3ListRead` dalam kebijakan ini memungkinkan Carlos melihat daftar semua bucket di Amazon S3. Pernyataan `AllowS3ProductionObjectActions` memungkinkan Carlos sepenuhnya mengakses objek di bucket `amzn-s3-demo-bucket-production`.

Selain itu, kebijakan berbasis sumber daya berikut (disebut kebijakan buket) diberlakukan untuk bucket `amzn-s3-demo-bucket-production` di akun 222222222222.

JSON

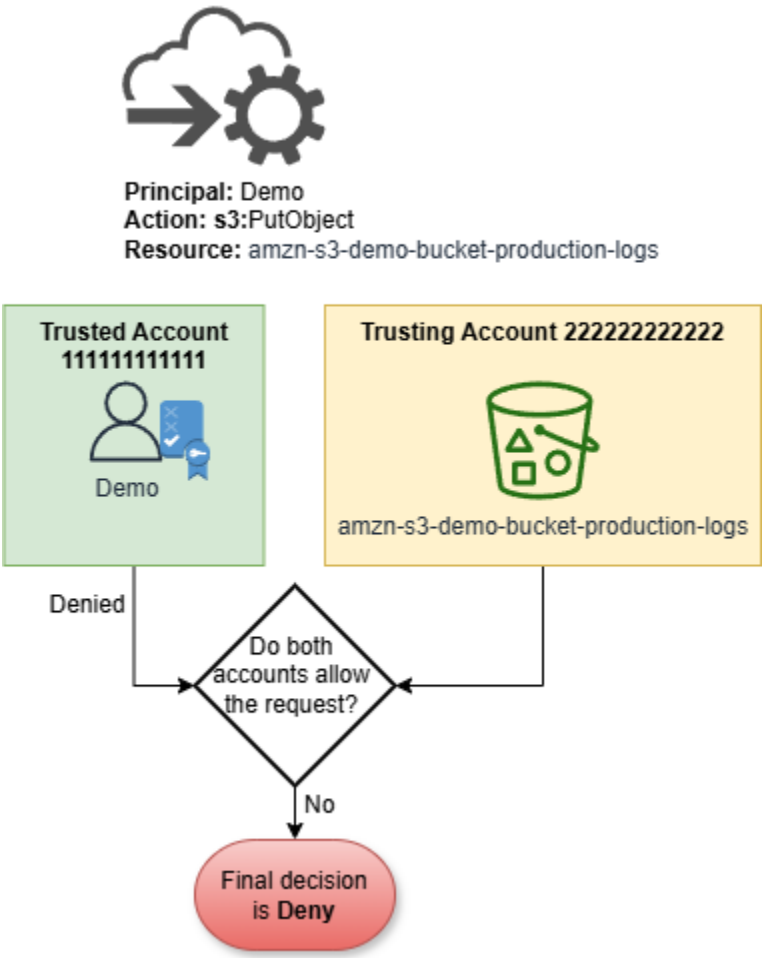
```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:role/Demo" },
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        }
    ]
}
```

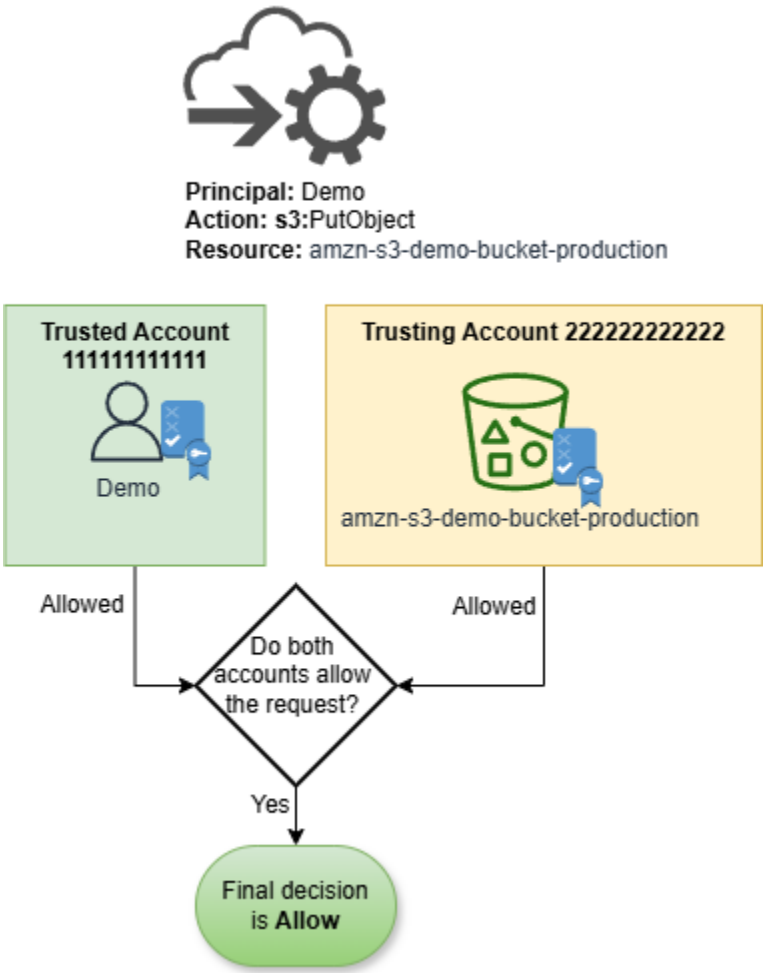
Kebijakan ini memungkinkan `Demo` peran untuk mengakses objek di `amzn-s3-demo-bucket-production` bucket. Peran dapat membuat dan mengedit, tetapi tidak menghapus objek di ember. Peran tidak dapat mengelola ember itu sendiri.

Ketika Carlos membuat permintaannya untuk menyimpan file ke `amzn-s3-demo-bucket-production-logs` ember, AWS tentukan kebijakan apa yang berlaku untuk permintaan tersebut. Dalam hal ini, kebijakan berbasis identitas yang melekat pada `Demo` peran adalah satu-satunya kebijakan yang berlaku di akun. `111111111111` Di akun `222222222222`, tidak ada kebijakan berbasis sumber daya yang dilampirkan ke bucket `amzn-s3-demo-bucket-production-logs`. Ketika AWS mengevaluasi akun`111111111111`, ia mengembalikan keputusan. `Deny` Ini karena pernyataan `DenyS3Logs` dalam kebijakan berbasis identitas secara eksplisit menolak akses ke bucket log. Untuk informasi selengkapnya tentang bagaimana permintaan dievaluasi di satu akun, lihat Bagaimana AWS logika kode penegakan mengevaluasi permintaan untuk mengizinkan atau menolak akses.

Karena permintaan tersebut secara tegas ditolak di dalam salah satu akun, keputusan akhir adalah menolak permintaan tersebut.



Asumsikan bahwa Carlos kemudian menyadari kesalahannya dan mencoba menyimpan file ke ember. `Production` AWS pertama memeriksa akun `111111111111` untuk menentukan apakah permintaan diizinkan. Hanya kebijakan berbasis identitas yang berlaku, dan memungkinkan permintaan. AWS kemudian memeriksa akun`222222222222`. Hanya kebijakan berbasis sumber daya yang dilampirkan pada bucket `Production` yang berlaku, dan ini mengizinkan permintaan. Karena kedua akun mengizinkan permintaan tersebut, keputusan akhir adalah untuk mengizinkan permintaan tersebut.



Perbedaan antara penolakan tegas dan implisit.

Permintaan menghasilkan penolakan eksplisit jika kebijakan yang berlaku mencakup pernyataan `Deny`. Jika kebijakan yang berlaku untuk permintaan mencakup pernyataan `Allow` dan `Deny`, pernyataan `Deny` mengalahkan pernyataan `Allow`. Permintaan ditolak secara tegas.

Penolakan implisit terjadi saat tidak ada pernyataan `Deny` yang berlaku, tetapi juga tidak ada pernyataan `Allow`. Karena prinsipal IAM ditolak aksesnya secara default, mereka harus secara eksplisit diizinkan untuk melakukan tindakan. Jika tidak, akse pengguna akan ditolak secara implisit.

Saat Anda merancang strategi otorisasi, Anda harus membuat kebijakan dengan pernyataan `Allow` agar prinsipal Anda berhasil membuat permintaan. Namun, Anda dapat memilih kombinasi penyangkalan eksplisit dan implisit.

Misalnya, Anda dapat membuat kebijakan berikut yang mencakup tindakan yang diizinkan, tindakan yang ditolak secara implisit, dan tindakan yang ditolak secara eksplisit. `AllowGetList`Pernyataan

ini memungkinkan akses hanya-baca ke tindakan IAM yang dimulai dengan awalan dan. `Get` `List` Semua tindakan lain di IAM, seperti ditolak `iam:CreatePolicy` secara implisit. `DenyReports`Pernyataan tersebut secara eksplisit menolak akses ke laporan IAM dengan menolak akses ke tindakan yang menyertakan akhiran, seperti. `Report` `iam:GetOrganizationsAccessReport` Jika seseorang menambahkan kebijakan lain ke prinsipal ini untuk memberi mereka akses ke laporan IAM, seperti`iam:GenerateCredentialReport`, permintaan terkait laporan masih ditolak karena penolakan eksplisit ini.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Tata bahasa kebijakan IAM JSON

Halaman ini menyajikan tata bahasa formal untuk bahasa yang digunakan untuk membuat kebijakan JSON di IAM. Kami menyajikan tata bahasa ini sehingga Anda dapat memahami cara menyusun dan memvalidasi kebijakan.

Untuk contoh kebijakan, lihat topik berikut:

- Kebijakan dan izin di AWS Identity and Access Management

- Contoh kebijakan berbasis identitas IAM
- Contoh Kebijakan untuk Bekerja di Konsol Amazon EC2 dan Contoh Kebijakan untuk Bekerja Dengan AWS CLI, Amazon EC2 CLI, atau AWS SDK di Panduan Pengguna Amazon EC2.
- Contoh Kebijakan Bucket dan Contoh Kebijakan Pengguna di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Untuk contoh kebijakan yang digunakan dalam AWS layanan lain, buka dokumentasi untuk layanan tersebut.

Topik

- Bahasa kebijakan dan JSON
- Konvensi yang digunakan dalam tata bahasa ini
- Tata Bahasa
- Catatan tata bahasa kebijakan

Bahasa kebijakan dan JSON

Kebijakan dinyatakan dalam JSON. Saat Anda membuat atau mengedit kebijakan JSON, IAM dapat melakukan validasi kebijakan untuk membantu Anda membuat kebijakan yang efektif. IAM mengidentifikasi kesalahan sintaks JSON, sementara IAM Access Analyzer menyediakan pemeriksaan kebijakan tambahan dengan rekomendasi untuk membantu Anda menyempurnakan kebijakan lebih lanjut. Untuk mempelajari selengkapnya tentang validasi kebijakan, lihat Validasi kebijakan IAM. Untuk mempelajari lebih lanjut tentang pemeriksaan kebijakan IAM Access Analyzer dan rekomendasi yang dapat ditindaklanjuti, lihat validasi kebijakan IAM Access Analyzer.

Dalam dokumen ini, kami tidak memberikan deskripsi lengkap mengenai segala sesuatu yang termasuk dalam JSON yang valid. Namun, berikut ini beberapa aturan dasar JSON:

- Spasi kosong antara entitas individu diperbolehkan.
- Nilai diapit dengan tanda petik. Tanda petik adalah bersifat opsional untuk nilai numerik dan Boolean.
- Banyak elemen (misalnya, `action_string_list` dan `resource_string_list`) dapat mengambil susunan JSON sebagai nilai. Susunan dapat mengambil satu atau beberapa nilai. Jika ada lebih dari satu nilai yang disertakan, susunan tersebut berada dalam kurung persegi (`[` dan `]`) dan dipisahkan dengan koma, seperti dalam contoh berikut:

```
"Action" : ["ec2:Describe*","ec2:List*"]
```

- Tipe data dasar JSON (Boolean, angka, dan string) ditentukan dalam RFC 7159.

Konvensi yang digunakan dalam tata bahasa ini

Konvensi berikut digunakan dalam tata bahasa ini:

- Karakter berikut adalah token JSON dan disertakan dalam kebijakan:

  ```
  { } [ ] " , :
  ```

- Karakter berikut adalah karakter khusus dalam tata bahasa dan tidak disertakan dalam kebijakan:

  ```
  = < > ( ) |
  ```

- Jika suatu elemen memungkinkan beberapa nilai, elemen tersebut ditunjukkan menggunakan nilai berulang, pemisah koma, dan elipsis (`...`). Contoh:

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Jika beberapa nilai diperbolehkan, memasukkan hanya satu nilai juga akan valid. Hanya untuk satu nilai, koma di akhir harus dihilangkan. Jika elemen mengambil sebuah susunan (ditandai dengan [dan]) tetapi hanya satu nilai yang disertakan, tanda kurung bersifat opsional. Contoh:

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```

- Tanda tanya (?) setelah elemen tertentu menunjukkan bahwa elemen tersebut bersifat opsional. Contoh:

  ```
  <version_block?>
  ```

  Namun, pastikan untuk merujuk pada catatan yang mengikuti daftar tata bahasa untuk perincian tentang elemen opsional.

- Garis vertikal (`|`) di antara elemen menunjukkan alternatif. Dalam tata bahasa, tanda kurung menentukan ruang lingkup alternatif. Contoh:

  ```
  ("Principal" | "NotPrincipal")
  ```

- Elemen yang harus berupa string literal diapit dengan tanda petik ganda ("). Contoh:

```
<version_block> = "Version" : ("2008-10-17" | "2012-10-17" )
```

Untuk catatan tambahan, lihat Catatan tata bahasa kebijakan mengikuti daftar tata bahasa.

Tata Bahasa

Daftar berikut menjelaskan tata bahasa kebijakan. Untuk konvensi yang digunakan dalam daftar, lihat bagian sebelumnya. Untuk informasi tambahan, lihat catatan setelahnya.

> **Note**
>
> Tata bahasa ini menjelaskan kebijakan yang ditandai dengan versi `2008-10-17` dan `2012-10-17` . Elemen kebijakan `Version` berbeda dari versi kebijakan. Elemen kebijakan `Version` digunakan dalam kebijakan dan menentukan versi bahasa kebijakan. Versi kebijakan, di sisi lain, dibuat ketika Anda membuat perubahan pada kebijakan yang dikelola pelanggan di IAM. Perubahan kebijakan tidak mengesampingkan kebijakan yang ada. Sebagai gantinya, IAM membuat versi baru dari kebijakan yang dikelola. Untuk mempelajari lebih lanjut tentang elemen kebijakan `Version`, lihat Elemen kebijakan IAM JSON: Version. Untuk mempelajari selengkapnya tentang versi kebijakan, lihat the section called "Peningkatan versi IAM".

```
policy  = {
     <version_block?>,
     <id_block?>,
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17"        | "2012-10-17"       )

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
```

```
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | <action_string> | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Catatan tata bahasa kebijakan

- Satu kebijakan dapat berisi serangkaian pernyataan.
- Kebijakan memiliki ukuran maksimum antara 2.048 karakter hingga 10.240 karakter, bergantung pada entitas yang menyertai kebijakan tersebut. Untuk informasi selengkapnya, lihat IAM dan AWS STS kuota. Perhitungan ukuran kebijakan tidak mencakup karakter spasi kosong.
- Elemen individu dilarang berisi beberapa kasus dari kunci yang sama. Misalnya, Anda tidak dapat menyertakan blok `Effect` dua kali dalam pernyataan yang sama.

- Blok dapat muncul dalam urutan apa pun. Misalnya, `version_block` dapat mengikuti `id_block` dalam kebijakan. Demikian pula, `effect_block`, `principal_block`, `action_block` dapat muncul dengan urutan apa pun dalam pernyataan.
- `id_block` bersifat opsional dalam kebijakan berbasis sumber daya. Komponen itu dilarang dimasukkan dalam kebijakan berbasis identitas.
- `principal_block` diperlukan dalam kebijakan berbasis sumber daya (misalnya, dalam kebijakan bucket Amazon S3 dan dalam kebijakan kepercayaan untuk peran IAM. Komponen itu dilarang dimasukkan dalam kebijakan berbasis identitas.
- Elemen `principal_map` di kebijakan bucket Amazon S3 dapat mencakup ID `CanonicalUser`. Sebagian besar kebijakan berbasis sumber daya tidak mendukung pemetaan ini. Untuk mempelajari lebih lanjut tentang menggunakan ID pengguna kanonik dalam kebijakan bucket, lihat Menentukan Prinsip dalam Kebijakan di Panduan Pengguna Layanan Peny impanan Sederhana Amazon.
- Setiap nilai string (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string`, dan versi string dari `condition_value`) dapat memiliki batasan panjang minimal dan maksimal, nilai tertentu yang diizinkan, atau format internal yang diperlukan.

Catatan tentang nilai string

Bagian ini menyediakan informasi tambahan tentang nilai string yang digunakan dalam elemen yang berbeda dalam kebijakan.

`action_string`

Terdiri dari namespace layanan, titik dua, dan nama tindakan Nama tindakan dapat mencakup kartu bebas. Contoh:

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"
```

```
"Action":[
  "s3:Get*",
  "s3:List*"
]
```

policy_id_string

Menyediakan cara untuk menyertakan informasi tentang kebijakan secara keseluruhan. Beberapa layanan, seperti Amazon SQS dan Amazon SNS, menggunakan elemen `Id` dengan cara khusus. Kecuali jika dibatasi oleh layanan individu, policy_id_string dapat mencakup spasi. Beberapa layanan mewajibkan nilai ini unik di dalam akun AWS .

> **Note**
>
> `id_block` diperbolehkan dalam kebijakan berbasis sumber daya, tetapi tidak dalam kebijakan berbasis identitas

Tidak ada batas panjang, meskipun string ini berkontribusi pada panjang kebijakan secara keseluruhan, yang dibatasi.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Memberikan cara untuk menyertakan informasi tentang pernyataan individu. Untuk kebijakan IAM, karakter alfanumerik dasar (A-Z, a-z,0-9) adalah satu-satunya karakter yang diizinkan dalam nilai. `Sid` Layanan AWS lain yang mendukung kebijakan sumber daya mungkin memiliki persyaratan lain untuk nilai `Sid`. Misalnya, beberapa layanan mengharuskan nilai ini menjadi unik dalam sebuah Akun AWS, dan beberapa layanan mengizinkan karakter tambahan seperti spasi dalam `Sid` nilai.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

`principal_id_string`

Menyediakan cara untuk menentukan prinsipal menggunakan Nama Sumber Daya Amazon (ARN) dari, pengguna IAM Akun AWS, peran IAM, pengguna federasi, atau pengguna peran yang diasumsikan. Untuk a Akun AWS, Anda juga dapat menggunakan formulir pendek al `AWS:accountnumber` ih-alih ARN lengkap. Untuk semua opsi termasuk layanan AWS , peran yang diasumsikan, dan sebagainya, lihat Cara menentukan prinsipal.

Perhatikan bahwa Anda dapat menggunakan * hanya untuk menentukan "everyone/anonymous." Anda tidak dapat menggunakannya untuk menentukan bagian dari nama atau ARN.

`resource_string`

Dalam sebagian besar kasus, terdiri atas Amazon Resource Name (ARN Anda dapat menggunakan wildcard (* dan?) dalam bagian sumber daya ARN. Untuk informasi selengkapnya tentang menggunakan wildcard di ARN, lihat. Menggunakan wildcard di jalur

> **Note**
>
> Bila Anda menentukan ARN yang tidak lengkap (satu dengan kurang dari enam bidang standar) dalam kebijakan berbasis identitas, AWS secara otomatis melengkapi ARN dengan menambahkan karakter wildcard (*) ke semua bidang yang hilang. Misalnya, menentukan `arn:aws:sqs` setara dengan`arn:aws:sqs:*:*:*`, yang memberikan akses ke semua sumber daya Amazon Amazon SQS di semua wilayah dan akun. Namun, kebijakan sesi yang diteruskan ke AWS STS AssumeRole,AssumeRoleWithWebIdentity, dan AssumeRoleWithSAML permintaan, tidak mendukung ARN yang tidak lengkap. Menggunakan ARN yang tidak lengkap dalam kebijakan sesi akan mengakibatkan `MalformedPolicyDocumentException` kesalahan.

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*"
```

`condition_type_string`

Mengidentifikasi tipe kondisi yang diuji, seperti `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, dll. Untuk daftar lengkap tipe kondisi, lihat Elemen kebijakan JSON IAM: Operator kondisi.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Mengidentifikasi kunci kondisi yang nilainya akan diuji untuk menentukan apakah kondisi terpenuhi. AWS mendefinisikan satu set kunci kondisi yang tersedia di semua AWS layanan, termasuk`aws:PrincipalType`,`aws:SecureTransport`, dan`aws:userid`.

Untuk daftar kunci AWS kondisi, lihatAWS kunci konteks kondisi global. Untuk kunci kondisi kepatuhan yang bersifat khusus terhadap layanan, lihat dokumentasi untuk layanan tersebut seperti berikut:

- Menentukan Kondisi dalam Kebijakan di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon
- Kebijakan IAM untuk Amazon EC2 di Panduan Pengguna Amazon EC2.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
```

```
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Mengidentifikasi nilai condition_key_string yang menentukan apakah kondisi terpenuhi. Untuk daftar lengkap nilai yang valid untuk jenis kondisi, lihatElemen kebijakan JSON IAM: Operator kondisi.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

AWS kebijakan terkelola untuk fungsi pekerjaan

Sebaiknya gunakan kebijakan yang memberikan hak istimewa paling sedikit, atau hanya memberikan izin yang diperlukan untuk melakukan tugas. Cara paling aman untuk memberikan hak istimewa paling sedikit adalah dengan menulis kebijakan khusus hanya dengan izin yang dibutuhkan oleh tim Anda. Anda harus membuat proses untuk memungkinkan tim Anda meminta lebih banyak izin bila diperlukan. Dibutuhkan waktu dan keahlian untuk membuat kebijakan yang dikelola pelanggan IAM yang hanya memberi tim Anda izin yang mereka butuhkan.

Untuk mulai menambahkan izin ke identitas IAM Anda (pengguna, grup pengguna, dan peran), Anda dapat menggunakanAWS kebijakan terkelola. AWS kebijakan yang dikelola mencakup kasus penggunaan umum dan tersedia di Anda Akun AWS. AWS kebijakan terkelola tidak memberikan sedikit izin hak istimewa. Anda harus mempertimbangkan risiko keamanan dengan memberi kepala sekolah Anda lebih banyak izin daripada yang mereka butuhkan untuk melakukan pekerjaan mereka.

Anda dapat melampirkan AWS kebijakan terkelola, termasuk fungsi pekerjaan, ke identitas IAM apa pun. Untuk beralih ke izin hak istimewa terkecil, Anda dapat menjalankan AWS Identity and

Access Management dan Access Analyzer untuk memantau prinsipal dengan kebijakan terkel AWS ola. Setelah mempelajari izin mana yang digunakan, Anda dapat menulis kebijakan khusus atau membuat kebijakan hanya dengan izin yang diperlukan untuk tim Anda. Ini kurang aman, tetapi memberikan lebih banyak fleksibilitas saat Anda mempelajari bagaimana tim Anda menggunakannya AWS.

AWS kebijakan yang dikelola untuk fungsi pekerjaan dirancang agar selaras dengan fungsi pekerjaan umum di industri TI. Anda dapat menggunakan kebijakan ini untuk memberikan izin yang diperlukan untuk melaksanakan tugas yang diharapkan dari seseorang dalam fungsi pekerjaan tertentu. Kebijakan ini mengonsolidasikan izin untuk banyak layanan ke dalam kebijakan tunggal yang lebih mudah digunakan daripada memiliki izin yang tersebar di banyak kebijakan.

Menggunakan Peran untuk Menggabungkan Layanan

Beberapa kebijakan menggunakan peran layanan IAM untuk membantu Anda memanfaatkan fitur yang ditemukan di AWS layanan lain. Kebijakan ini memberikan akses ke`iam:passrole`, yang memungkinkan pengguna dengan kebijakan untuk meneruskan peran ke AWS layanan. Peran ini mendelegasikan izin IAM ke AWS layanan untuk melakukan tindakan atas nama Anda.

Anda harus membuat peran sesuai dengan kebutuhan Anda. Misalnya, kebijakan Administrator Jaringan mengizinkan pengguna dengan kebijakan untuk meneruskan peran bernama “flow-logs-vpc” ke layanan Amazon. CloudWatch CloudWatch menggunakan peran itu untuk mencatat dan menangkap lalu lintas IP untuk VPC yang dibuat oleh pengguna.

Untuk mengikuti praktik terbaik keamanan, kebijakan untuk fungsi pekerjaan mencakup filter yang membatasi nama peran valid yang dapat diberikan. Tindakan ini membantu menghindari memberikan izin yang tidak perlu. Jika pengguna Anda memang memerlukan peran layanan opsional, Anda harus membuat peran yang mengikuti konvensi penamaan yang ditetapkan dalam kebijakan. Kemudian, Anda memberikan izin untuk peran tersebut. Setelah selesai, pengguna dapat mengonfigurasi layanan untuk menggunakan peran, memberikan izin apa pun yang diberikan oleh peran.

Di bagian berikut, nama setiap kebijakan adalah tautan ke halaman perincian kebijakan di Konsol Manajemen AWS. Di sana Anda dapat melihat dokumen kebijakan dan meninjau izin yang diberikan.

Fungsi pekerjaan administrator

AWS nama kebijakan yang dikelola: AdministratorAccess

Kasus penggunaan: Pengguna ini memiliki akses penuh dan dapat mendelegasikan izin untuk setiap layanan dan sumber daya di AWS.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan semua tindakan untuk semua AWS layanan dan untuk semua sumber daya di akun. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat AdministratorAccess Panduan Refer AWS ensi Kebijakan Terkelola.

> **Note**
>
> Sebelum pengguna atau peran IAM dapat mengakses AWS Manajemen Penagihan dan Biaya konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan pengguna IAM dan akses peran. Untuk melakukannya, ikuti petunjuk di Berikan akses ke konsol penagihan untuk mendelegasikan akses ke konsol penagihan.

Fungsi pekerjaan penagihan

AWS nama kebijakan terkelola: Pen agihan

Kasus penggunaan: Pengguna ini perlu melihat informasi penagihan, menyiapkan pembayaran, dan mengotorisasi pembayaran. Pengguna dapat memantau akumulasi biaya untuk seluruh AWS layanan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin penuh untuk mengelola penagihan, biaya, metode pembayaran, anggaran, dan laporan. Untuk contoh kebijakan manajemen biaya tambahan, lihat contoh AWS Billing kebijakan di Panduan AWS Manajemen Penagihan dan Biaya Pengguna. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat Pen agihan di Panduan AWS Referensi Kebijakan Terkelola.

> **Note**
>
> Sebelum pengguna atau peran IAM dapat mengakses AWS Manajemen Penagihan dan Biaya konsol dengan izin dalam kebijakan ini, Anda harus terlebih dahulu mengaktifkan

pengguna IAM dan akses peran. Untuk melakukannya, ikuti petunjuk di Berikan akses ke konsol penagihan untuk mendelegasikan akses ke konsol penagihan.

Fungsi pekerjaan administrator basis data

AWS nama kebijakan yang dikelola: DatabaseAdministrator

Kasus penggunaan: Pengguna ini mengatur, mengonfigurasi, dan memelihara database di AWS Cloud.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat, mengonfigurasi, dan memelihara basis data. Ini termasuk akses ke layanan AWS database, seperti Amazon DynamoDB, Amazon Relational Database Service (RDS), dan Amazon Redshift. Lihat kebijakan untuk mengetahui daftar lengkap layanan basis data yang mendukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat DatabaseAdministrator Panduan Refer AWS ensi Kebijakan Terkelola.

Kebijakan fungsi pekerjaan ini mendukung kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini mengizinkan tindakan `iam:PassRole` hanya untuk peran yang disebutkan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan ter AWS kelola ini
Memungkinkan pengguna memantau basis data RDS	peran pemantauan rds	Peran Amazon RDS untuk Pemantauan yang Ditingkatkan	AmazonRDS EnhancedMonitoring Role
Izinkan AWS Lambda untuk memantau	rdbms-lambda-akses	Amazon EC2	AWS Lambda_FullAccess

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan ter AWS kelola ini
database Anda dan mengakses database eksternal			
Izinkan Lambda mengunggah file ke Amazon S3 dan ke cluster Amazon Redshift dengan DynamoDB	lambda_exec_role	AWS Lambda	Membuat kebijakan pengelolaan baru sebagaimana yang ditentukan dalam Blog Big Data AWS
Izinkan fungsi Lambda bertindak sebagai pemicu untuk tabel DynamoDB Anda	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Memungkinkan fungsi Lambda mengakses Amazon RDS dalam VPC	lambda-vpc-eksekusi-peran	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan Peng AWS Lambda embang	AWSLambda VPCAccessExecutionRole
Izinkan AWS Data Pipeline untuk mengakses AWS sumber daya Anda	DataPipelineDefaultRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan Peng AWS Data Pipeline embang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat peran IAM untuk AWS Data Pipeline

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	Pilih kebijakan ter AWS kelola ini
Izinkan aplikasi Anda yang berjalan di instans Amazon EC2 untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan Peng AWS Data Pipeline embang	AmazonEC2RoleforDataPipelineRole

Fungsi pekerjaan ilmuwan data

AWS nama kebijakan yang dikelola: DataScientist

Kasus penggunaan: Pengguna ini menjalankan pekerjaan dan kueri Hadoop. Pengguna juga mengakses dan menganalisis informasi untuk analitik data dan kecerdasan bisnis.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat, mengelola, dan menjalankan kueri di cluster Amazon EMR dan melakukan analitik data dengan alat seperti Amazon QuickSight. Kebijakan ini mencakup akses ke layanan ilmuwan data tambahan, seperti Amazon EC2 AWS Data Pipeline, Amazon Kinesis, Amazon Machine Learning, dan SageMaker AI. Lihat kebijakan untuk mengetahui daftar lengkap layanan ilmuwan data yang mendukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat DataScientist Panduan Refer AWS ensi Kebijakan Terkelola.

Kebijakan fungsi pekerjaan ini mendukung kemampuan untuk meneruskan peran ke AWS layanan. Satu pernyataan memungkinkan meneruskan peran apa pun ke SageMaker AI. Pernyataan lainnya mengizinkan tindakan `iam:PassRole` hanya untuk peran yang disebutkan dalam tabel berikut.

Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan yang dikelola untuk memilih
Memungkinkan instans Amazon EC2 mengakses layanan dan sumber daya yang sesuai untuk klaster	EMR-EC2_DefaultRole	Amazon EMR for EC2	AmazonElasticMapReduceforEC2Role
Memungkinkan akses Amazon EMR untuk mengakses layanan dan sumber daya Amazon EC2 untuk klaster	EMR_ DefaultRole	Amazon EMR	AmazonEMRServicePolicy_v2
Izinkan Kinesis Managed Service untuk Apache Flink mengakses sumber data streaming	Kinesis*	Membuat peran dengan kebijakan kepercayaan sebagaimana yang ditentukan dalam Blog Big Data AWS	Lihat Blog Big Data AWS yang menguraikan empat kemungkinan opsi, bergantung pada kasus penggunaan Anda
Izinkan AWS Data Pipeline untuk mengakses AWS sumber daya Anda	DataPipelineDefaultRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan Peng AWS Data Pipeline embang	AWS Data Pipeline Dokumentasi mencantumkan izin yang diperlukan untuk kasus penggunaan ini. Lihat peran IAM untuk AWS Data Pipeline

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan yang dikelola untuk memilih
Izinkan aplikasi Anda yang berjalan di instans Amazon EC2 untuk mengakses sumber daya Anda AWS	DataPipelineDefaultResourceRole	Membuat peran dengan kebijakan kepercayaan seperti yang didefinisikan dalam Panduan Peng AWS Data Pipeline embang	AmazonEC2RoleforDataPipelineRole

Fungsi pekerjaan pengguna daya developer

AWS nama kebijakan yang dikelola: PowerUserAccess

Kasus penggunaan: Pengguna ini melakukan tugas pengembangan aplikasi dan dapat membuat dan mengonfigurasi sumber daya dan layanan yang mendukung pengembangan aplikasi AWS sadar.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Pernyataan pertama kebijakan ini menggunakan `NotAction` elemen untuk mengizinkan semua tindakan untuk semua AWS layanan dan untuk semua sumber daya kecuali AWS Identity and Access Management, AWS Organizations, dan AWS Account Management. Pernyataan kedua memberikan izin IAM untuk membuat peran tertaut layanan. Ini diwajibkan oleh beberapa layanan yang harus mengakses sumber daya di layanan lain, seperti bucket Amazon S3.

Ini juga memberikan AWS Organizations izin untuk melihat informasi tentang organisasi pengguna, termasuk email akun manajemen dan batasan organisasi. Meskipun kebijakan ini membatasi IAM AWS Organizations, kebijakan ini memungkinkan pengguna untuk melakukan semua tindakan Pusat Identitas IAM jika Pusat Identitas IAM diaktifkan. Ini juga memberikan izin Manajemen Akun untuk melihat Wil AWS ayah mana yang diaktifkan atau dinonaktifkan untuk akun.

Fungsi pekerjaan administrator jaringan

AWS nama kebijakan yang dikelola: NetworkAdministrator

Kasus penggunaan: Pengguna ini ditugaskan untuk mengatur dan memelihara sumber daya AWS jaringan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memelihara sumber daya jaringan di Auto Scaling, Amazon EC2, AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing, AWS Elastic Beanstalk, Amazon SNS, CloudWatch, CloudWatch Logs, Amazon S3, IAM, dan Amazon Virtual Private Cloud. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat NetworkAdministrator Panduan Refer AWS ensi Kebijakan Terkelola.

Fungsi pekerjaan ini membutuhkan kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini memberikan `iam:GetRole` dan `iam:PassRole` hanya untuk peran yang ditentukan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan yang dikelola untuk memilih
Memungkinkan Amazon VPC membuat dan mengelola CloudWatch log di Log atas nama pengguna untuk memantau lalu lintas IP yang masuk dan keluar dari VPC Anda	log-alir*	Membuat peran dengan kebijakan kepercayaan sebagaimana yang ditetapkan dalam Panduan Pengguna VPC Amazon	Kasus penggunaan ini tidak memiliki kebijakan terkel AWS ola yang ada, tetapi dokumentasi mencantumkan izin yang diperlukan. Lihat Panduan Pengguna VPC Amazon.

Read-only akses

AWS nama kebijakan yang dikelola: ReadOnlyAccess

Kasus penggunaan: Pengguna ini memerlukan akses baca-saja ke setiap sumber daya dalam. Akun AWS

> **Important**
>
> Pengguna ini juga akan memiliki akses untuk membaca data dalam layanan penyimpanan seperti bucket Amazon S3 dan tabel Amazon DynamoDB.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi tugas, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan:Kebijakan ini memberikan izin untuk daftar, mendapatkan, menguraikan, dan sebaliknya melihat sumber daya dan atribut mereka. Ini tidak termasuk fungsi bermutasi seperti membuat atau menghapus. Kebijakan ini mencakup akses baca-saja ke AWS layanan terkait keamanan, seperti dan. AWS Identity and Access Management AWS Manajemen Penagihan dan Biaya

Lihat kebijakan untuk daftar lengkap layanan dan tindakan yang didukung kebijakan ini. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat ReadOnlyAccess Panduan Refer AWS ensi Kebijakan Terkelola. Jika Anda memerlukan kebijakan serupa yang tidak memberikan akses untuk membaca data di layanan penyimpanan, lihatView-only fungsi pekerjaan pengguna.

Tindakan layanan MCP akses penuh

AWS nama kebijakan yang dikelola: AWSMcpServiceActionsFullAccess

Kasus penggunaan: Pengguna ini memerlukan akses ke AWS layanan menggunakan server AWS MCP. Kebijakan ini tidak memberikan akses ke tindakan yang diambil oleh layanan MCP ke AWS layanan lain.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi

selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk memanggil tindakan layanan AWS MCP apa pun. Anda dapat menggunakan ketika Anda tidak perlu menentukan izin per layanan AWS MCP. Itu tidak memberikan izin untuk tindakan yang diambil oleh layanan MCP ke AWS layanan lain, izin tersebut harus selalu diberikan secara terpisah dan sebagai tambahan untuk tindakan layanan MCP. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat AWSMcpServiceActionsFullAccess Panduan Refer AWS ensi Kebijakan Terkelola.

Fungsi pekerjaan auditor keamanan

AWS nama kebijakan yang dikelola: SecurityAudit

Kasus penggunaan: Pengguna ini memantau akun agar mematuhi persyaratan keamanan. Pengguna ini dapat mengakses catatan dan peristiwa untuk menginvestigasi kemungkinan adanya pelanggaran keamanan atau kemungkinan adanya aktivitas berbahaya.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk melihat data konfigurasi untuk banyak AWS layanan dan untuk meninjau log mereka. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat SecurityAudit Panduan Refer AWS ensi Kebijakan Terkelola.

Fungsi pekerjaan pengguna

AWS nama kebijakan yang dikelola: AWSSupportAccess

Kasus penggunaan: Pengguna ini AWS menghubungi Dukungan, membuat kasus dukungan, dan melihat status kasus yang ada.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memperbarui Dukungan kasus. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat AWSSupportAccess Panduan Refer AWS ensi Kebijakan Terkelola.

Fungsi pekerjaan administrator sistem

AWS nama kebijakan yang dikelola: SystemAdministrator

Kasus penggunaan: Pengguna ini mengatur dan memelihara sumber daya untuk operasi pengembangan.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini memberikan izin untuk membuat dan memelihara sumber daya di berbagai AWS layanan, termasuk, Amazon AWS CloudTrail, CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config AWS Directory Service, Amazon EC2,, AWS Identity and Access Management AWS Key Management Service, Amazon RDS AWS Lambda, Route 53, Amazon S3, Amazon SES, Amazon SQS, AWS Trusted Advisor, dan Amazon VPC. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat SystemAdministrator Panduan Refer AWS ensi Kebijakan Terkelola.

Fungsi pekerjaan ini membutuhkan kemampuan untuk meneruskan peran ke AWS layanan. Kebijakan ini memberikan `iam:GetRole` dan `iam:PassRole` hanya untuk peran yang ditentukan dalam tabel berikut. Untuk informasi lebih lanjut, lihat Menciptakan peran dan memberlakukan kebijakan (konsol) dalam topik ini. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan yang dikelola untuk memilih
Memungkinkan aplikasi berjalan dalam instans EC2 di klaster Amazon ECS untuk mengakses Amazon ECS	ecr-sysadmin-*	Peran Amazon EC2 untuk Layanan EC2 Container	AmazonEC2 ContainerServicefo rEC2Role

Kasus penggunaan	Nama peran (* adalah wildcard)	Jenis peran layanan untuk dipilih	AWS kebijakan yang dikelola untuk memilih
Memungkinkan pengguna untuk memantau basis data	peran pemantauan rds	Peran Amazon RDS untuk Pemantauan yang Ditingkatkan	AmazonRDS EnhancedMonitoringRole
Izinkan aplikasi yang berjalan di instans EC2 untuk mengakses AWS sumber daya.	ec2-sysadmin-*	Amazon EC2	Contoh kebijakan untuk peran yang memberikan akses ke bucket S3 seperti yang ditunjukkan dalam Panduan Pengguna Amazon EC2; sesuaikan sesuai kebutuhan
Izinkan Lambda membaca aliran DynamoDB dan menulis ke Log CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambda DynamoDBExecutionRole

View-only fungsi pekerjaan pengguna

AWS nama kebijakan yang dikelola: ViewOnlyAccess

Kasus penggunaan: Pengguna ini dapat melihat daftar sumber AWS daya dan metadata dasar di akun di seluruh layanan. Pengguna tidak dapat membaca konten sumber daya atau metadata yang melampaui kuota dan informasi daftar sumber daya.

Pembaruan kebijakan: AWS memelihara dan memperbarui kebijakan ini. Untuk riwayat perubahan kebijakan ini, lihat kebijakan di konsol IAM, lalu pilih tab Versi kebijakan. Untuk informasi selengkapnya tentang pembaruan kebijakan fungsi pekerjaan, lihat Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan.

Deskripsi kebijakan: Kebijakan ini `List*` memberikan`Describe*`,, `Get*View*`, dan `Lookup*` akses ke sumber daya untuk AWS layanan. Untuk melihat tindakan apa yang tercakup dalam kebijakan ini untuk setiap layanan, lihat ViewOnlyAccess. Untuk informasi selengkapnya tentang kebijakan terkelola, lihat ViewOnlyAccess Panduan Refer AWS ensi Kebijakan Terkelola.

Pembaruan untuk AWS kebijakan terkelola untuk fungsi pekerjaan

Semua kebijakan ini dikelola oleh AWS dan terus diperbarui untuk menyertakan dukungan untuk layanan baru dan kemampuan baru saat ditambahkan oleh AWS layanan. Kebijakan ini tidak dapat dimodifikasi oleh pelanggan. Anda dapat membuat salinan kebijakan dan kemudian memodifikasi salinannya, tetapi salinan tersebut tidak diperbarui secara otomatis karena AWS memperkenalkan layanan baru dan operasi API.

Untuk kebijakan fungsi pekerjaan, Anda dapat melihat riwayat versi serta waktu dan tanggal setiap pembaruan di konsol IAM. Untuk melakukannya, gunakan tautan di halaman ini untuk melihat detail kebijakan. Lalu pilih tab Versi kebijakan untuk melihat versi. Halaman ini menunjukkan 25 versi terakhir dari sebuah kebijakan. Untuk melihat semua versi kebijakan, panggil AWS CLI perintah get-policy-version atau operasi API. GetPolicyVersion

> **Note**
>
> Anda dapat memiliki hingga lima versi kebijakan yang dikelola pelanggan, tetapi tetap AWS mempertahankan riwayat versi lengkap kebijakan ter AWS kelola.

Menciptakan peran dan memberlakukan kebijakan (konsol)

Beberapa kebijakan yang tercantum sebelumnya memberikan kemampuan untuk mengonfigurasi AWS layanan dengan peran yang memungkinkan layanan tersebut melakukan operasi atas nama Anda. Kebijakan fungsi pekerjaan menentukan nama peran yang tepat yang harus Anda gunakan atau setidaknya menyertakan awalan yang menentukan bagian pertama dari nama yang dapat digunakan. Untuk membuat salah satu peran ini, lakukan langkah-langkah dalam prosedur berikut.

Untuk membuat peran untuk Layanan AWS (Konsol IAM)

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
3. Untuk Ti pe entitas tepercaya, pilih Layanan AWS.

4. Untuk Layanan atau kasus penggunaan, pilih layanan, lalu pilih kasus penggunaan. Kasus penggunaan ditentukan oleh layanan untuk menyertakan kebijakan kepercayaan yang diperlukan layanan.
5. Pilih Berikutnya.
6. Untuk kebijakan Iz in, opsi tergantung pada kasus penggunaan yang Anda pilih:
   - Jika layanan menentukan izin untuk peran, Anda tidak dapat memilih kebijakan izin.
   - Pilih dari kumpulan kebijakan izin terbatas.
   - Pilih dari semua kebijakan izin.
   - Pilih kebijakan tanpa izin, buat kebijakan setelah peran dibuat, lalu lampirkan kebijakan ke peran tersebut.
7. (Opsional) Tetapkan batas izin. Ini adalah fitur lanjutan yang tersedia untuk peran layanan, tetapi bukan peran tertaut layanan.
   a. Buka bagian Tetapkan batas izin, lalu pilih Gunakan batas izin untuk mengontrol izin peran maksimum.

      IAM menyertakan daftar kebijakan yang AWS dikelola dan dikelola pelanggan di akun Anda.
   b. Pilih kebijakan yang akan digunakan untuk batas izin.
8. Pilih Berikutnya.
9. Untuk nama Per an, opsi tergantung pada layanan:
   - Jika layanan menentukan nama peran, Anda tidak dapat mengedit nama peran.
   - Jika layanan mendefinisikan awalan untuk nama peran, Anda dapat memasukkan akhiran opsional.
   - Jika layanan tidak menentukan nama peran, Anda dapat memberi nama peran tersebut.

> **Important**
>
> Saat Anda memberi nama peran, perhatikan hal berikut:
>
> - Nama peran harus unik dalam nama Anda Akun AWS, dan tidak dapat dibuat unik berdasarkan kasus.
>
>   Misalnya, jangan membuat peran bernama keduanya **`PRODROLE`** dan**`prodrole`**. Ketika nama peran digunakan dalam kebijakan atau sebagai bagian dari ARN, nama peran tersebut peka huruf besar/kecil, namun ketika nama peran muncul kepada

pelanggan di konsol, seperti selama proses masuk, nama peran tidak peka huruf besar/kecil.

- Anda tidak dapat mengedit nama peran setelah dibuat karena entitas lain mungkin mereferensikan peran tersebut.

10. (Opsional) Untuk Deskripsi, masukkan deskripsi untuk peran.
11. (Opsional) Untuk mengedit kasus penggunaan dan izin untuk peran, di bagian Langkah 1: Pilih entitas tepercaya atau Langkah 2: Tambahkan izin, pilih Edit.
12. (Opsional) Untuk membantu mengidentifikasi, mengatur, atau mencari peran, tambahkan tag sebagai pasangan kunci-nilai. Untuk informasi selengkapnya tentang menggunakan tag di IAM, lihat Tag untuk AWS Identity and Access Management sumber daya di Panduan Pengguna IAM.
13. Tinjau peran lalu pilih Buat peran.

Contoh 1: Mengonfigurasi pengguna sebagai administrator basis data (konsol)

Contoh ini menunjukkan langkah-langkah yang diperlukan untuk mengonfigurasi Alice, pengguna IAM, sebagai Administrator Basis Data. Anda menggunakan informasi di baris pertama tabel di bagian tersebut dan memungkinkan pengguna mengaktifkan pemantauan Amazon RDS. Anda melampirkan DatabaseAdministrator kebijakan ke pengguna IAM Alice sehingga mereka dapat mengelola layanan database Amazon. Kebijakan itu juga memungkinkan Alice untuk meneruskan peran yang di `rds-monitoring-role` panggil ke layanan Amazon RDS yang memungkinkan layanan untuk memantau database Amazon RDS atas nama mereka.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Kebijakan, ketik **`database`** di kotak pencarian, lalu tekan enter.
3. Pilih tombol radio untuk DatabaseAdministrator kebijakan, pilih Tind akan, lalu pilih Lamp irkan.
4. Dalam daftar entitas, pilih Alice lalu pilih Lamp irkan kebijakan. Alice sekarang dapat mengelola AWS database. Namun, agar Alice dapat memantau basis data tersebut, Anda harus mengonfigurasi peran layanan.
5. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
6. Pilih jenis peran AWS Layanan, lalu pilih Amazon RDS.
7. Pilih kasus penggunaan Peran Amazon RDS untuk Pemantauan yang Ditingkatkan.

8. Amazon RDS mendefinisikan izin untuk peran Anda. Pilih Next: Review (Berikutnya: Tinjauan) untuk melanjutkan.
9. Nama peran harus salah satu dari yang ditentukan oleh DatabaseAdministrator kebijakan yang sekarang dimiliki Alice. Salah satunya adalah **`rds-monitoring-role`**. Masukkan itu untuk nama Role.
10. (Opsional) Untuk Deskripsi peran, masukkan deskripsi.
11. Setelah meninjau perinciannya, pilih Buat peran.
12. Sekarang Alice dapat mengaktifkan RDS Enhanced Monitoring dalam bagian Pemantauan di konsol Amazon RDS. Misalnya, mereka mungkin melakukan ini ketika mereka membuat instance DB, membuat replika baca, atau memodifikasi instance DB. Mereka harus memasukkan nama peran yang mereka buat (rds-monitoring-role) di kotak Peran Pemantauan saat mereka menyetel Akti fkan Pemantauan yang Ditingkatkan ke Ya.

Contoh 2: Mengonfigurasi pengguna sebagai administrator jaringan (konsol)

Contoh ini menunjukkan langkah-langkah yang diperlukan untuk mengkonfigurasi Jorge, pengguna IAM, sebagai Administrator ??? Jaringan. Ini menggunakan informasi dalam tabel di bagian itu untuk memungkinkan Jorge memantau lalu lintas IP yang pergi ke dan dari VPC. Ini juga memungkinkan Jorge untuk menangkap informasi itu di log di CloudWatch Log. Anda melampirkan NetworkAdministrator kebijakan ke pengguna IAM Jorge sehingga mereka dapat mengonfigurasi sumber daya AWS jaringan. Kebijakan itu juga memungkinkan Jorge untuk meneruskan peran yang namanya dimulai dengan `flow-logs*` ke Amazon EC2 saat Anda membuat log alur. Dalam skenario ini, tidak seperti Contoh 1, tidak ada jenis peran layanan yang ditentukan sebelumnya sehingga Anda harus melakukan beberapa langkah secara berbeda.

1. Masuk ke Konsol Manajemen AWS dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan lalu masukkan **`network`** di kotak pencarian, lalu tekan enter.
3. Pilih tombol radio di sebelah NetworkAdministrator kebijakan, pilih Tind akan, lalu pilih Lamp irkan.
4. Dalam daftar pengguna, pilih kotak centang di sebelah Jorge dan kemudian pilih Lamp irkan kebijakan. Jorge sekarang dapat mengelola sumber daya AWS jaringan. Namun, untuk mengaktifkan pemantauan lalu lintas IP di VPC, Anda harus mengonfigurasi peran layanan.

5. Karena peran layanan yang perlu Anda buat tidak memiliki kebijakan yang dikelola sebelumnya, Anda harus membuatnya terlebih dahulu. Pada panel navigasi, pilih Kebijakan, lalu pilih Buat kebijakan.
6. Di bagian Editor kebijakan, pilih opsi JSON dan salin teks dari dokumen kebijakan JSON berikut. Tempel teks ini ke kotak teks JSON.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": [
       {
         "Action": [
           "logs:CreateLogGroup",
           "logs:CreateLogStream",
           "logs:PutLogEvents",
           "logs:DescribeLogGroups",
           "logs:DescribeLogStreams"
         ],
         "Effect": "Allow",
         "Resource": "*"
       }
     ]
   }
   ```

7. Selesaikan peringatan keamanan, kesalahan, atau peringatan umum yang dihasilkan selama validasi kebijakan, lalu pilih Berikutnya.

   > **Note**
   >
   > Anda dapat beralih antara opsi editor Visual dan JSON kapan saja. Namun, jika Anda melakukan perubahan atau memilih Berikutnya di editor Visual, IAM dapat merestrukturisasi kebijakan Anda untuk mengoptimalkannya bagi editor visual. Untuk informasi selengkapnya, lihat Restrukturisasi kebijakan.

8. Pada halaman Tinjau dan buat, **`vpc-flow-logs-policy-for-service-role`** ketik nama kebijakan. Tinjau Izin yang ditentukan dalam kebijakan ini untuk melihat izin yang diberikan oleh kebijakan Anda, lalu pilih Buat kebijakan untuk menyimpan pekerjaan Anda.

   Kebijakan baru muncul dalam daftar kebijakan terkelola dan siap diberlakukan.

9. Di panel navigasi konsol IAM, pilih Peran, dan lalu pilih Buat peran.
10. Pilih jenis peran AWS Layanan, lalu pilih Amazon EC2.
11. Pilih kasus penggunaan Amazon EC2.
12. Di halaman Berlakukan kebijakan izin, pilih kebijakan yang telah Anda buat sebelumnya, vpc-flow-logs-policy-for-service-role, lalu pilih Berikutnya: Tinjauan.
13. Nama peran harus diizinkan oleh NetworkAdministrator kebijakan yang sekarang dimiliki Jorge. Nama apa pun yang dimulai dengan `flow-logs-` diperbolehkan. Untuk contoh ini, **`flow-logs-for-jorge`** masukkan nama Peran.
14. (Opsional) Untuk Deskripsi peran, masukkan deskripsi.
15. Setelah meninjau perinciannya, pilih Buat peran.
16. Sekarang, Anda dapat mengonfigurasi kebijakan kepercayaan yang diperlukan untuk skenario ini. Pada halaman Per an, pilih peran flow-logs-for-jorge (nama, bukan kotak centang). Pada halaman perincian peran baru Anda, pilih Hubungan kepercayaan, lalu pilihEdit hubungan kepercayaan.
17. Ubah baris "Layanan" agar dibaca sebagai berikut, menggantikan entri untuk `ec2.amazonaws.com`:

    ```
            "Service": "vpc-flow-logs.amazonaws.com"
    ```

18. Jorge sekarang dapat membuat log aliran untuk VPC atau subnet di konsol Amazon EC2. Saat Anda membuat log alur, tentukan peran flow-log s-for-jorge. Peran tersebut memiliki izin untuk membuat data log dan menuliskan data ke log itu.

AWS kunci konteks kondisi global

Ketika kepala sekolah membuat permintaan ke AWS, AWS mengumpulkan informasi permintaan ke dalam konteks ??? permintaan. Anda dapat menggunakan elemen `Condition` dari kebijakan JSON untuk membandingkan kunci dalam konteks permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda. Informasi permintaan disediakan oleh sumber yang berbeda, termasuk prinsipal yang membuat permintaan, sumber daya yang diminta, dan metadata tentang permintaan itu sendiri.

Kunci kondisi global dapat digunakan di semua AWS layanan. Meskipun kunci kondisi ini dapat digunakan di semua kebijakan, kunci tidak tersedia di setiap konteks permintaan. Misalnya, kunci `aws:SourceAccount` kondisi hanya tersedia ketika panggilan ke sumber daya Anda dilakukan

langsung oleh prinsi AWS pal layanan. Untuk mempelajari lebih lanjut tentang keadaan di mana kunci global disertakan dalam konteks permintaan, lihat Informasi Ketersediaan untuk setiap kunci.

Beberapa layanan individual membuat kunci kondisi mereka sendiri yang tersedia dalam konteks permintaan untuk layanan lain. Cross-service kunci kondisi adalah jenis kunci kondisi global yang menyertakan awalan yang cocok dengan nama layanan, seperti `ec2:` atau`lambda:`, tetapi tersedia di layanan lain.

Service-specific kunci kondisi didefinisikan untuk digunakan dengan AWS layanan individual. Misalnya, Amazon S3 memungkinkan Anda menulis kebijakan dengan kunci `s3:VersionId` kondisi untuk membatasi akses ke versi tertentu dari objek Amazon S3. Kunci kondisi ini unik untuk layanan, artinya hanya berfungsi dengan permintaan ke layanan Amazon S3. Untuk kunci kondisi khusus layanan, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk Layanan dan pilih AWS layanan yang kuncinya ingin Anda lihat.

> **Note**
>
> Jika Anda menggunakan kunci kondisi yang hanya tersedia dalam beberapa keadaan, Anda dapat menggunakan versi IfExists dari operator kondisi. Jika kunci kondisi hilang dari konteks permintaan, kebijakan dapat gagal dalam evaluasi. Misalnya, gunakan blok berikut dengan operator `...IfExists` untuk mencocokkan saat permintaan berasal dari rentang IP tertentu atau dari VPC tertentu. Jika salah satu atau kedua kunci tidak disertakan dalam konteks permintaan, kondisi masih akan mengembalikan `true`. Nilai hanya diperiksa jika kunci yang ditentukan disertakan dalam konteks permintaan. Untuk informasi selengkapnya tentang cara kebijakan dievaluasi ketika kunci tidak ada untuk operator lain, lihat Oper ator kondisi.
>
> ```
> "Condition": {
>     "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
>     "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
> }
> ```

> **Important**
>
> Untuk membandingkan kondisi Anda terhadap konteks permintaan dengan beberapa nilai kunci, Anda harus menggunakan operator kumpulan `ForAllValues` atau `ForAnyValue`. Gunakan operator set hanya dengan kunci kondisi multivalue. Jangan gunakan operator set

dengan kunci syarat bernilai tunggal. Untuk informasi selengkapnya, lihat Mengatur operator untuk kunci konteks multinilai.

Properti prinsipal	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: PrincipalArn aws: PrincipalAccount aws: PrincipalOrgPaths aws: PrincipalOrg ID aw:PrincipalTag/ *kunci tag* aws: PrincipalIsAWSService aws: PrincipalServiceName	aw: AssumedRoot aw: FederatedProvider aw: TokenIssueTime aw: SignInSessionArn aw: MultiFactorAuthAge aw: MultiFactorAuthPresent aws: ChatbotSourceArn aws: Ec2Instan	aws: SourceIp aws: SourceVpc aws: SourceVpcArn aws: SourceVpce aws: VpceAccount aws: VpceOrgID aws: VpceOrgPaths aw: VpcSourceIp	aw: ResourceAccount aws: ResourceOrgID aw: ResourceOrgPaths aw:ResourceTag/ *kunci tag*	aws: CalledVia aws: CalledViaFirst aw: CalledViaLast aws: CalledViaAWSMCP aws:ViaAWSService AWS:melalui AWSMCPService aws: CurrentTime aw: EpochTime aws:referer

Properti prinsipal	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
aws: PrincipalServiceNamesList aws: PrincipalType aws:userid aws:username	ceSourceVpc aws: Ec2InstanceSourcePrivateIPv4 aws: SourceIdentity aw: ViaCustomerDomain ec2: RoleDelivery ec2: SourceInstanceArn lem: RoleAssumedBy lem: CredentialIssuingService pembuatankode: BuildArn			aw: RequestedRegion aw:RequestTag/ *kunci tag* aws: TagKeys aws: SecureTransport aw: SourceAccount aw: SourceArn aws: SourceOrg ID aw: SourceOrgPaths aws: UserAgent aw: IsMcpServiceAction

Properti prinsipal	Properti sesi peran	Properti jaringan	Properti dari sumber daya	Properti permintaan
	pembuatan kode: ProjectArn Lambda: SourceFunctionArn SSM: SourceInstanceArn toko identitas: UserId			

Kunci kondisi sensitif

Kunci kondisi berikut dianggap sensitif. Penggunaan wildcard dalam kunci kondisi ini tidak memiliki kasus penggunaan yang valid, bahkan dengan substring nilai kunci dengan wildcard. Ini karena wildcard mungkin cocok dengan kunci kondisi dengan nilai apa pun, yang dapat menimbulkan risiko keamanan.

- aws: PrincipalAccount
- aws: PrincipalOrg ID
- aw: ResourceAccount
- aws: ResourceOrg ID
- aw: SourceAccount
- aws: SourceOrg ID
- aws: SourceVpc
- aws: SourceVpce
- aws: VpceAccount
- aws: VpceOrg ID

Properti prinsipal

Gunakan kunci kondisi berikut untuk membandingkan detail tentang prinsipal yang membuat permintaan dengan properti utama yang Anda tentukan dalam kebijakan. Untuk daftar kepala sekolah yang dapat membuat permintaan, lihatCara menentukan prinsipal.

aws: PrincipalArn

Gunakan kunci ini untuk membandingkan Amazon Resource Name (ARN) prinsipal yang melakukan permintaan layanan ke layanan dengan ARN yang Anda tentukan dalam kebijakan. Untuk peran IAM, konteks permintaan memberikan ARN peran, bukan ARN pengguna yang mengambil peran tersebut.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan untuk semua permintaan yang ditandatangani. Permintaan anonim tidak menyertakan kunci ini. Anda dapat menentukan jenis prinsipal berikut dalam kunci kondisi ini:
  - IAM Role
  - Pengguna IAM
  - AWS STS prinsipal pengguna federasi
  - Akun AWS pengguna root
- Tipe data — ARN

  AWS merekomendasikan agar Anda menggunakan operator ARN al ih-alih operator string saat membandingkan ARN.
- Jenis nilai — Single-valued
- Contoh nilai Daftar berikut menunjukkan nilai konteks permintaan yang dikembalikan untuk berbagai jenis prinsipal yang dapat Anda tentukan dalam kunci `aws:PrincipalArn` kondisi:
  - Peran IAM — Konteks permintaan berisi nilai berikut untuk kunci `aws:PrincipalArn` kondisi. Jangan tentukan sesi peran yang diasumsikan ARN sebagai nilai untuk kunci kondisi ini. Untuk informasi selengkapnya tentang prinsipal sesi peran yang diasumsikan, lihatPrinsip sesi peran.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - Pengguna IAM — Konteks permintaan berisi nilai berikut untuk kunci `aws:PrincipalArn` kondisi.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```

- AWS STS prinsipal pengguna federasi — Konteks permintaan berisi nilai berikut untuk kunci kondisi. `aws:PrincipalArn`

```
arn:aws:sts::123456789012:federated-user/user-name
```

- Akun AWS pengguna root — Konteks permintaan berisi nilai berikut untuk kunci kondisi`aws:PrincipalArn`. Ketika Anda menentukan pengguna root ARN sebagai nilai untuk kunci `aws:PrincipalArn` kondisi, itu membatasi izin hanya untuk pengguna root dari Akun AWS. Ini berbeda dengan menentukan pengguna root ARN dalam elemen utama kebijakan berbasis sumber daya, yang mendelegasikan otoritas ke. Akun AWS Untuk informasi selengkapnya tentang menentukan ARN pengguna root dalam elemen utama kebijakan berbasis sumber daya, lihat. Akun AWS utama

```
arn:aws:iam::123456789012:root
```

Anda dapat menentukan ARN pengguna root sebagai nilai untuk kunci kondisi `aws:PrincipalArn` dalam kebijakan kontrol AWS Organizations layanan (SCP). SCP adalah jenis kebijakan organisasi yang digunakan untuk mengelola izin di organisasi Anda dan hanya memengaruhi akun anggota di organisasi. SCP membatasi izin untuk pengguna dan peran IAM dalam akun anggota, termasuk pengguna akun anggota. Untuk informasi selengkapnya tentang efek SCP pada izin, lihat Efek SCP pada izin di Panduan AWS Organizations Pengguna.

aws: PrincipalAccount

Gunakan kunci ini untuk membandingkan akun yang memiliki prinsipal yang meminta dengan pengenal akun yang Anda tentukan dalam kebijakan. Untuk permintaan anonim, konteks permintaan kembali`anonymous`.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data — String
- Jenis nilai — Single-valued

Dalam contoh berikut, akses ditolak kecuali untuk kepala sekolah dengan nomor `123456789012` akun.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws: PrincipalOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur untuk kepala sekolah yang membuat permintaan ke jalur dalam kebijakan. Prinsip tersebut dapat berupa pengguna IAM, peran IAM, prinsipal pengguna AWS STS federasi, atau. Pengguna root akun AWS Dalam kebijakan, kunci kondisi ini memastikan pemohon adalah anggota akun dalam akar organisasi atau unit organisasi (OU) yang ditentukan di AWS Organizations. Jal AWS Organizations ur adalah representasi teks dari struktur AWS Organizations entitas. Untuk informasi lebih lanjut tentang cara menggunakan dan memahami jalur, lihat Memahami AWS Organizations jalur entitas.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal merupakan anggota dari suatu organisasi. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data — String (daftar)
- Jenis nilai — Multivalue

> **Note**
>
> ID organisasi unik secara global, tetapi ID OU dan ID akar hanya unik dalam organisasinya. Ini berarti bahwa tidak ada dua organisasi yang memiliki ID organisasi yang sama. Namun, organisasi lain mungkin memiliki OU atau root dengan ID yang sama seperti milik Anda. Kami merekomendasikan agar Anda selalu menyertakan ID organisasi saat menentukan OU atau root.

Misalnya, kondisi berikut dikembalikan `true` untuk prinsipal di akun yang dilampirkan langsung ke `ou-ab12-22222222` OU, tetapi tidak di OU turunannya.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

Kondisi berikut dikembalikan `true` untuk prinsipal dalam akun yang dilampirkan langsung ke OU atau OU turunannya. Ketika Anda menyertakan wildcard, Anda harus menggunakan operator kondisi `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

Kondisi berikut dikembalikan `true` untuk kepala sekolah dalam akun yang dilampirkan langsung ke salah satu OU anak, tetapi tidak langsung ke OU induk. Kondisi sebelumnya ditujukan untuk OU atau turunannya. Kondisi berikut hanya ditujukan untuk turunannya (dan turunan dari turunan tersebut).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

Kondisi berikut memungkinkan akses untuk setiap prinsipal dalam organisasi `o-a1b2c3d4e5`, terlepas dari OU indukannya.

```
"Condition" : { "ForAnyValue:StringLike" : {
```

```
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` adalah kunci kondisi multinilai. Kunci multivalue dapat memiliki beberapa nilai dalam konteks permintaan. Saat Anda menggunakan beberapa nilai dengan operator kondisi `ForAnyValue`, jalur prinsipal harus sesuai dengan salah satu jalur yang tercantum dalam kebijakan. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Mengatur operator untuk kunci konteks multinilai.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws: PrincipalOrg ID

Gunakan kunci ini untuk membandingkan pengidentifikasi organisasi tempat AWS Organizations kepala sekolah yang meminta berada dengan pengidentifikasi yang ditentukan dalam kebijakan.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal merupakan anggota dari suatu organisasi. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data — String
- Jenis nilai — Single-valued

Kunci global ini menyediakan alternatif untuk mendaftar semua ID akun untuk semua akun AWS di organisasi. Anda dapat menggunakan tombol kondisi ini untuk menyederhanakan penentuan elemen `Principal` dalam kebijakan berbasis sumber daya. Anda dapat menentukan ID organisasi dalam elemen kondisi. Saat Anda menambah dan menghapus akun, kebijakan yang mencakup kunci `aws:PrincipalOrgID` akan secara otomatis menyertakan akun yang benar dan tidak memerlukan pembaruan manual.

Misalnya, kebijakan bucket Amazon S3 berikut memungkinkan anggota akun mana pun di organisasi `o-xxxxxxxxxxx` untuk menambahkan objek ke dalam bucket `amzn-s3-demo-bucket`.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxx"}
   }
 }
}
```

> **Note**
>
> Kondisi global ini juga berlaku untuk akun manajemen suatu AWS organisasi. Kebijakan ini mencegah semua kepala sekolah di luar organisasi tertentu mengakses bucket Amazon S3. Ini termasuk AWS layanan apa pun yang berinteraksi dengan sumber daya internal Anda, seperti AWS CloudTrail mengirim data log ke bucket Amazon S3 Anda. Untuk mempelajari cara memberikan akses AWS layanan dengan aman, lihataws: PrincipalIsAWSService.

Untuk informasi selengkapnya AWS Organizations, lihat Apa itu AWS Organizations? dalam Panduan AWS Organizations Pengguna.

aw:PrincipalTag/kunci tag

Gunakan kunci ini untuk membandingkan tanda yang dilampirkan ke prinsipal yang mengajukan permintaan dengan tanda yang Anda tentukan dalam kebijakan. Jika prinsipal memiliki lebih dari satu tag yang disematkan, konteks permintaan mencakup satu kunci `aws:PrincipalTag` untuk setiap kunci tag yang disematkan.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan jika prinsipal menggunakan pengguna IAM dengan tag yang disematkan. Ini disertakan untuk prinsipal yang menggunakan peran IAM dengan tag yang disematkan atau tag sesi. Permintaan anonim tidak menyertakan kunci ini.

- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menambahkan atribut khusus ke pengguna atau peran dalam bentuk pasangan nilai kunci. Untuk informasi selengkapnya tentang tag IAM, lihat Tag untuk AWS Identity and Access Management sumber daya. Anda dapat menggunakan `aws:PrincipalTag` untuk mengontrol akses untuk prinsipal AWS .

Contoh ini menunjukkan cara membuat kebijakan berbasis identitas yang memungkinkan pengguna dengan **`department=hr`** tag mengelola pengguna, grup, atau peran IAM. Untuk menggunakan kebijakan ini, ganti kebijakan *`italicized placeholder text`* dalam contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau ubah kebijakan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:Get*",
        "iam:List*",
        "iam:Generate*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws: PrincipalIsAWSService

Gunakan kunci ini untuk memeriksa apakah panggilan ke sumber daya Anda dilakukan langsung oleh kepala AWS sekolah layanan. Misalnya, AWS CloudTrail menggunakan prinsip layanan

`cloudtrail.amazonaws.com` untuk menulis log ke bucket Amazon S3 Anda. Kunci konteks permintaan diatur ke benar ketika layanan menggunakan prinsipal layanan untuk melakukan tindakan langsung pada sumber daya Anda. Kunci konteks disetel ke false jika layanan menggunakan kredenSIAL dari prinsipal IAM untuk membuat permintaan atas nama kepala sekolah. Ini juga disetel ke false jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.

- Ketersediaan — Kunci ini hadir dalam konteks permintaan untuk semua permintaan API yang ditandatangani yang menggunakan AWS kredenSIAL. Permintaan anonim tidak menyertakan kunci ini.
- Tipe data — Boolean
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke identitas tepercaya dan lokasi jaringan yang diharapkan sambil memberikan akses ke AWS layanan dengan aman.

Dalam contoh kebijakan bucket Amazon S3 berikut, akses ke bucket dibatasi kecuali permintaan berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
```

```
  ]
}
```

Dalam video berikut, pelajari selengkapnya tentang cara menggunakan kunci kondisi `aws:PrincipalIsAWSService` dalam kebijakan.

Berikan akses dengan aman ke pengguna resmi Anda, lokasi jaringan yang diharapkan, dan AWS layanan bersama-sama.

aws: PrincipalServiceName

Gunakan kunci ini untuk membandingkan nama prinsipal layanan dalam kebijakan dengan prinsipal layanan yang membuat permintaan untuk sumber daya Anda. Anda dapat menggunakan kunci ini untuk memeriksa apakah panggilan ini dibuat oleh prinsipal layanan tertentu. Ketika prinsipal layanan membuat permintaan langsung ke sumber daya Anda,kunci `aws:PrincipalServiceName`berisi nama prinsipal layanan. Misalnya, nama utama AWS CloudTrail layanan adalah`cloudtrail.amazonaws.com`.

- Ketersediaan — Kunci ini hadir dalam permintaan saat panggilan dilakukan oleh kepala AWS sekolah layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
  - Jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.
  - Jika layanan menggunakan kredenSIAL prinsipal IAM untuk membuat permintaan atas nama kepala sekolah.
  - Jika panggilan dilakukan langsung oleh kepala sekolah IAM.
  - Jika panggilan dilakukan oleh pemohon anonim.
- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke identitas tepercaya dan lokasi jaringan yang diharapkan sambil memberikan akses ke AWS layanan dengan aman.

Dalam contoh kebijakan bucket Amazon S3 berikut, akses ke bucket dibatasi kecuali permintaan berasal dari `vpc-111bbb22` atau berasal dari prinsipal layanan, seperti CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

aws: PrincipalServiceNamesList

Kunci ini menyediakan daftar semua nama prinsipal layanan yang termasuk dalam layanan. Ini adalah kunci kondisi lanjutan. Anda dapat menggunakannya untuk membatasi layanan dari mengakses sumber daya Anda hanya dari Wilayah tertentu. Beberapa layanan mungkin membuat prinsipal layanan Regional untuk menunjukkan contoh tertentu dari layanan dalam Wilayah tertentu. Anda dapat membatasi akses ke sumber daya untuk instans layanan tertentu. Saat prinsipal layanan membuat permintaan langsung ke sumber daya Anda, `aws:PrincipalServiceNamesList` berisi daftar seluruh nama prinsipal layanan yang tidak diurutkan yang terkait dengan instans Regional layanan.

- Ketersediaan — Kunci ini hadir dalam permintaan saat panggilan dilakukan oleh kepala AWS sekolah layanan. Kunci ini tidak ada dalam situasi lain, termasuk berikut ini:
  - Jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah.
  - Jika layanan menggunakan kredenSIAL prinsipal IAM untuk membuat permintaan atas nama kepala sekolah.

  - Jika panggilan dilakukan langsung oleh kepala sekolah IAM.
  - Jika panggilan dilakukan oleh pemohon anonim.
- Tipe data — String (daftar)
- Jenis nilai — Multivalue

`aws:PrincipalServiceNamesList` adalah kunci kondisi multinilai. Kunci multivalue dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan operator `ForAnyValue` atau `ForAllValues` set dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Mengatur operator untuk kunci konteks multinilai.

aws: PrincipalType

Gunakan kunci ini untuk membandingkan tipe prinsipal yang membuat permintaan dengan tipe prinsipal yang Anda sebutkan dalam kebijakan. Untuk informasi selengkapnya, lihat Cara menentukan prinsipal. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data — String
- Jenis nilai — Single-valued

aws:userid

Gunakan kunci ini untuk membandingkan pengenal prinsipal pemohon dengan ID yang Anda tentukan dalam kebijakan. Untuk pengguna IAM, nilai konteks permintaan adalah ID pengguna. Untuk peran IAM, format nilai ini dapat bervariasi. Untuk detail tentang bagaimana informasi tersebut muncul untuk prinsipal yang berbeda, lihat Cara menentukan prinsipal. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk semua permintaan, termasuk permintaan anonim.
- Tipe data — String
- Jenis nilai — Single-valued

aws:username

Gunakan kunci ini untuk membandingkan nama pengguna pemohon dengan nama pengguna yang Anda sebutkan dalam kebijakan. Untuk detail tentang bagaimana informasi tersebut muncul untuk prinsipal yang berbeda, lihat Cara menentukan prinsipal. Untuk contoh spesifik dari nilai-nilai `principal` kunci, lihatNilai-nilai kunci utama.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan untuk pengguna IAM. Permintaan dan permintaan anonim yang dibuat menggunakan peran Pengguna root akun AWS atau IAM tidak menyertakan kunci ini. Permintaan yang dibuat menggunakan kredentif Pusat Identitas IAM tidak menyertakan kunci ini dalam konteks.
- Tipe data — String
- Jenis nilai — Single-valued

Properti sesi peran

Gunakan kunci kondisi berikut untuk membandingkan properti sesi peran pada saat sesi dibuat. Kunci kondisi ini hanya tersedia ketika permintaan dibuat oleh kepala sekolah dengan sesi peran atau kredentif utama pengguna federasi. Nilai untuk kunci kondisi ini disematkan dalam token sesi peran.

Peran adalah jenis prinsipal. Anda juga dapat menggunakan kunci kondisi dari Properti prinsipal bagian untuk mengevaluasi properti peran saat peran membuat permintaan.

aw: AssumedRoot

Gunakan kunci ini untuk memeriksa apakah permintaan dibuat menggunakan AssumeRoot. `AssumeRoot`mengembalikan kredentif jangka pendek untuk sesi pengguna root istimewa yang dapat Anda gunakan untuk mengambil tindakan istimewa pada akun anggota di organisasi Anda. Untuk informasi selengkapnya, lihat Mengelola akses root secara terpusat untuk akun anggota.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika kepala sekolah menggunakan kredenSIAL AssumeRoot untuk membuat permintaan.
- Tipe data — Boolean
- Jenis nilai — Single-valued

Dalam contoh berikut, ketika digunakan sebagai kebijakan kontrol layanan, menolak penggunaan kredenSIAL jangka panjang pengguna root di akun AWS Organizations anggota. Kebijakan ini tidak menolak `AssumeRoot` sesi melakukan tindakan yang diizinkan oleh `AssumeRoot` sesi.

JSON

```
{
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Deny",
          "Action":"*",
          "Resource": "*",
          "Condition":{
             "ArnLike":{
                "aws:PrincipalArn":[
                   "arn:aws:iam::*:root"
                ]
             },
             "Null":{
                "aws:AssumedRoot":"true"
             }
          }
       }
    ]
 }
```

aw: FederatedProvider

Gunakan kunci ini untuk membandingkan penyedia identitas penerbit (IdP) kepala sekolah dengan IdP yang Anda tentukan dalam kebijakan. Ini berarti bahwa peran IAM diasumsikan menggunakan `AssumeRoleWithWebIdentity` AWS STS operasi. Ketika kredentif sementara sesi peran yang dihasilkan digunakan untuk membuat permintaan, konteks permintaan mengidentifikasi IdP yang mengautentikasi identitas federasi asli.

- Ketersediaan — Kunci ini hadir dalam sesi peran peran yang diasumsikan menggunakan penyedia OpenID Connect (OIDC), dan dalam kebijakan kepercayaan peran saat penyedia OIDC digunakan untuk memanggil. `AssumeRoleWithWebIdentity`
- Tipe data — String *

- Jenis nilai — Single-valued

* Jenis data tergantung pada IdP Anda:

- Jika Anda menggunakan AWS IdP bawaan, seperti Amazon Cognito, nilai kuncinya akan berupa string. Nilai kuncinya mungkin terlihat seperti:`cognito-identity.amazonaws.com`.
- Jika Anda menggunakan IdP yang tidak terintegrasi ke AWS, seperti GitHub atau Amazon EKS, nilai kuncinya adalah AR N. Nilai kuncinya mungkin terlihat seperti:`arn:aws:iam::`*`111122223333`*`:oidc-provider/oidc.eks.`*`region`*`.amazonaws.com/id/`*`OIDC_Provider_ID`*.

Untuk informasi lebih lanjut tentang eksternal IdPs dan`AssumeRoleWithWebIdentity`, lihatSkenario umum. Untuk informasi selengkapnya, lihat Prinsip sesi peran.

aw: TokenIssueTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu saat kredensial keamanan sementara dikeluarkan dengan tanggal dan waktu yang Anda tentukan dalam kebijakan.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kunci tidak ada dalam permintaan AWS CLI, AWS API, atau AWS SDK yang dibuat menggunakan kunci akses.
- Tipe data — Tanggal
- Jenis nilai — Single-valued

Untuk mempelajari layanan mana yang mendukung penggunaan kredensial sementara, lihat AWS layanan yang bekerja dengan IAM.

aw: SignInSessionArn

Gunakan tombol ini untuk membandingkan AWS Sign-In OAuth sesi Anda dengan sesi yang Anda tentukan dalam kebijakan.

Saat Anda menggunakan alur OAuth berbasis seperti AWS CLI login (`aws login`) atau AWS MCP Server, ser AWS Sign-In takan sesi masuk ARN dalam kredensional yang dikeluarkan. AWS Sign-In menyebarkan ARN itu ke permintaan berikutnya yang dibuat menggunakan token OAuth akses. Dengan kunci ini, Anda dapat menghubungkan aktivitas API dengan sesi masuk asal dan menerapkan kebijakan IAM ke sesi individual.

Untuk alur OAuth interaktif, token penyegaran menentukan siklus hidup ARN sesi masuk. Token akses baru yang dikeluarkan selama penyegaran terus membawa sesi masuk yang sama ARN.

Untuk alur non- OAuth interaktif, token akses menentukan siklus hidup ARN sesi masuk. Setiap token akses baru berisi sesi masuk baru ARN.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat permintaan berasal dari AWS Sign-In OAuth sesi. Kunci ini tidak tersedia untuk sesi konsol.
- Tipe data — ARN
- Jenis nilai — Single-valued

Gunakan operator ARN alih-alih operator string saat Anda membandingkan ARN.

Kebijakan berikut menolak semua tindakan yang terkait dengan AWS Sign-In sesi tertentu, meskipun permintaan dari sesi aktif lainnya terus berlanjut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenySpecificSignInSession",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SignInSessionArn": "arn:aws:signin:us-east-1:111122223333:session/
session-id"
        }
      }
    }
  ]
}
```

aw: MultiFactorAuthAge

Gunakan kunci ini untuk membandingkan jumlah detik sejak prinsipal yang meminta diberi kewenangan menggunakan MFA dengan nomor yang Anda sebutkan dalam kebijakan. Untuk informasi selengkapnya tentang MFA, lihat AWS Multi-factor otentikasi di IAM.

> **Important**
>
> Kunci kondisi ini tidak ada untuk identitas federasi atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani permintaan AWS CLI, AWS API, atau AWS SDK. Untuk mempelajari selengkapnya tentang menambahkan perlindungan MFA ke operasi API dengan kredenSIAL keamanan sementara, lihatAkses API aman dengan MFA. Untuk memeriksa apakah MFA digunakan untuk memvalidasi identitas federasi IAM, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke AWS sebagai tag sesi. Lihat perinciannya di Lewati tag sesi di AWS STS. Untuk menerapkan MFA untuk identitas Pusat Identitas IAM, Anda dapat mengaktifkan atribut untuk kontrol akses untuk meneruskan klaim pernyataan SAML dengan metode otentikasi dari penyedia identitas Anda ke Pusat Identitas IAM.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika kepala sekolah menggunakan kredenSIAL keamanan sementara untuk membuat permintaan. Kebijakan dengan kondisi MFA dapat dilampirkan ke:
  - Pengguna atau grup IAM
  - Sumber daya seperti bucket Amazon S3, antrean Amazon Amazon SQS, atau topik Amazon SNS
  - Kebijakan kepercayaan dari peran IAM yang dapat diasumsikan oleh pengguna
- Tipe data — Numerik
- Jenis nilai — Single-valued

aw: MultiFactorAuthPresent

Gunakan kunci ini untuk memeriksa apakah otentikasi multi-faktor (MFA) digunakan untuk memvalidasi kreden tif keamanan sementara yang membuat permintaan.

> **Important**
>
> Kunci kondisi ini tidak ada untuk identitas federasi atau permintaan yang dibuat menggunakan kunci akses untuk menandatangani permintaan AWS CLI, AWS API, atau AWS SDK. Untuk mempelajari selengkapnya tentang menambahkan perlindungan MFA ke operasi API dengan kredenSIAL keamanan sementara, lihatAkses API aman dengan MFA.

Untuk memeriksa apakah MFA digunakan untuk memvalidasi identitas federasi IAM, Anda dapat meneruskan metode otentikasi dari penyedia identitas Anda ke AWS sebagai tag sesi. Lihat perinciannya di Lewati tag sesi di AWS STS. Untuk menerapkan MFA untuk identitas Pusat Identitas IAM, Anda dapat mengaktifkan atribut untuk kontrol akses untuk meneruskan klaim pernyataan SAML dengan metode otentikasi dari penyedia identitas Anda ke Pusat Identitas IAM.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan hanya jika prinsipal menggunakan kredensial sementara untuk membuat permintaan. Kebijakan dengan kondisi MFA dapat dilampirkan ke:
  - Pengguna atau grup IAM
  - Sumber daya seperti bucket Amazon S3, antrean Amazon Amazon SQS, atau topik Amazon SNS
  - Kebijakan kepercayaan dari peran IAM yang dapat diasumsikan oleh pengguna
- Tipe data — Boolean
- Jenis nilai — Single-valued

KredenSIAL sementara digunakan untuk mengotentikasi peran IAM dan pengguna IAM dengan token sementara dari AssumeRole atau GetSessionToken, dan pengguna. Konsol Manajemen AWS

Kunci akses pengguna IAM adalah kredenSIAL jangka panjang, tetapi dalam beberapa kasus, AWS membuat kredenSIAL sementara atas nama pengguna IAM untuk melakukan operasi. Dalam kasus ini, kunci `aws:MultiFactorAuthPresent` tersedia dalam permintaan dan diatur ke nilai `false`. Ada dua kasus umum hal ini bisa terjadi:

- Pengguna IAM secara Konsol Manajemen AWS tidak sadar menggunakan kredenSIAL sementara. Pengguna masuk ke konsol menggunakan nama pengguna dan kata sandi yang merupakan kredensial jangka panjang. Namun, di latar belakang, konsol membuat kredensial sementara atas nama pengguna.
- Jika pengguna IAM melakukan panggilan ke AWS layanan, layanan menggunakan kembali kredentif pengguna untuk membuat permintaan lain ke layanan lain. Misalnya, saat memanggil Athena untuk mengakses bucket Amazon S3, atau saat menggunakan CloudFormation untuk membuat instans Amazon EC2. Untuk permintaan berikutnya, gunakan kre AWS denSIAL sementara.

Untuk mempelajari layanan mana yang mendukung penggunaan kredensial sementara, lihat AWS layanan yang bekerja dengan IAM.

Kunci `aws:MultiFactorAuthPresent` tidak tersedia ketika perintah API atau CLI disebut kredensial jangka panjang, seperti pasangan access key pengguna. Oleh karena itu, kami merekomendasikan bahwa ketika Anda memeriksa kunci ini, Anda menggunakan versi `...IfExists` dari operator kondisi.

Penting untuk dipahami bahwa elemen `Condition` bukan merupakan suatu cara yang andal untuk memastikan apakah permintaan diautentikasi menggunakan MFA.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi dari efek `Deny`, elemen `Bool`, dan nilai `false` menolak permintaan yang dapat diautentikasi menggunakan MFA, tetapi tidak. Ini hanya berlaku untuk kredensial sementara yang mendukung penggunaan MFA. Pernyataan ini tidak menolak akses ke permintaan yang dilakukan menggunakan kredensial jangka panjang, atau permintaan yang diautentikasi menggunakan MFA. Gunakan contoh ini dengan hati-hati karena logikanya rumit dan tidak menguji apakah MFA-authentication benar-benar digunakan.

Jangan juga menggunakan kombinasi efek `Deny`, elemen `Null`, dan `true` karena contoh berperilaku dengan cara yang sama dan logikanya bahkan lebih rumit.

Kombinasi yang Direkomendasikan

Kami sarankan Anda menggunakan operator `BoolIfExists` untuk memastikan permintaan teraotentikasi menggunakan MFA ataukah tidak.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi dari `Deny`, `BoolIfExists`, dan `false` menolak permintaan yang tidak diautentikasi menggunakan MFA. Secara khusus, operator ini menolak permintaan dari kredensial sementara yang tidak menyertakan MFA. Ini juga menolak permintaan yang dibuat menggunakan kredenSIAL jangka panjang, seperti AWS CLI atau operasi AWS API yang dibuat menggunakan kunci akses. Operator `*IfExists` memeriksa keberadaan kunci `aws:MultiFactorAuthPresent` dan apakah bisa tersedia atau tidak, sebagaimana yang ditunjukkan oleh keberadaannya. Gunakan ini saat Anda

ingin menolak permintaan yang tidak diautentikasi menggunakan MFA. Ini lebih aman, tetapi dapat merusak kode atau skrip apa pun yang menggunakan kunci akses untuk mengakses AWS API AWS CLI atau.

Kombinasi Alternatif

Anda juga dapat menggunakan `BoolIfExists` operator untuk mengizinkan MFA-authenticated permintaan AWS CLI dan/atau permintaan AWS API yang dibuat menggunakan kredenSIAL jangka panjang.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Kondisi ini cocok jika kunci ada dan tersedia atau jika kunci tidak ada. Kombinasi dari `Allow`, `BoolIfExists`, dan `true` mengaktifkan permintaan yang diautentikasi menggunakan MFA, atau permintaan yang tidak dapat diautentikasi menggunakan MFA. Ini berarti bahwa AWS CLI, operasi AWS API, dan AWS SDK diizinkan ketika pemohon menggunakan kunci akses jangka panjangnya. Kombinasi ini tidak memungkinkan permintaan dari kredensial sementara yang bisa, tetapi tidak menyertakan MFA.

Saat Anda membuat kebijakan menggunakan editor visual konsol IAM dan memilih MFA yang diperlukan, kombinasi ini akan diterapkan. Pengaturan ini memerlukan MFA untuk akses konsol, tetapi memungkinkan akses terprogram tanpa MFA.

Atau, Anda dapat menggunakan operator `Bool` hanya untuk permintaan terprogram dan konsol saat diautentikasi menggunakan MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Kombinasi ini dari`Allow`,`Bool`, dan hanya `true` mengizinkan MFA-authenticated permintaan. Ini hanya berlaku untuk kredensial sementara yang mendukung penggunaan MFA. Pernyataan ini tidak memungkinkan akses ke permintaan yang dilakukan menggunakan access key jangka panjang, atau untuk permintaan yang dilakukan menggunakan kredensial sementara tanpa MFA.

Jangan gunakan susunan kebijakan yang serupa dengan yang berikut ini untuk memastikan adanya kunci MFA atau tidak:

```
#####   WARNING: USE WITH CAUTION   #####
```

```
"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Kombinasi dari efek `Allow`, elemen `Null`, dan nilai `false` hanya memungkinkan permintaan yang dapat diautentikasi menggunakan MFA, terlepas dari apakah permintaan tersebut benar-benar diautentikasi. Ini memungkinkan semua permintaan yang dibuat menggunakan kredensial sementara, dan menolak akses untuk kredensial jangka panjang. Gunakan contoh ini dengan hati-hati karena tidak menguji apakah MFA-authentication benar-benar digunakan.

aws: ChatbotSourceArn

Gunakan kunci ini untuk membandingkan konfigurasi obrolan sumber ARN yang ditetapkan oleh prinsipal dengan ARN konfigurasi obrolan yang Anda tentukan dalam kebijakan peran IAM yang terkait dengan konfigurasi saluran Anda. Anda dapat mengotorisasi permintaan berdasarkan sesi peran asumsi yang diprakarsai oleh Amazon Q Developer di aplikasi obrolan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan oleh Pengembang Amazon Q dalam layanan aplikasi obrolan setiap kali sesi peran diasumsikan. Nilai kuncinya adalah konfigurasi obrolan ARN, seperti ketika Anda menjalankan AWS CLI perintah dari saluran obrolan.
- Tipe data — ARN
- Jenis nilai — Single-valued
- Nilai contoh — `arn:aws::chatbot::123456789021:chat-configuration/slack-channel/private_channel`

Kebijakan berikut menolak permintaan menempatkan Amazon S3 pada bucket yang ditentukan untuk semua permintaan yang berasal dari saluran Slack.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleS3Deny",
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
```

```
            "Condition": {
                "ArnLike": {
                      "aws:ChatbotSourceArn": "arn:aws:chatbot::*:chat-
configuration/slack-channel/*"
                }
            }
        }
    ]
}
```

aws: Ec2InstanceSourceVpc

Kunci ini mengidentifikasi VPC tempat kredentif peran Amazon EC2 IAM dikirimkan. Anda dapat menggunakan kunci ini dalam kebijakan dengan kunci `aws:SourceVPC` global untuk memeriksa apakah panggilan dibuat dari VPC (`aws:SourceVPC`) yang cocok dengan VPC tempat kredensia dikirimkan ke (). `aws:Ec2InstanceSourceVpc`

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali pemohon menandatangani permintaan dengan kredensi peran Amazon EC2. Ini dapat digunakan dalam kebijakan IAM, kebijakan kontrol layanan, kebijakan titik akhir VPC, dan kebijakan sumber daya.
- Tipe data — String
- Jenis nilai — Single-valued

Kunci ini dapat digunakan dengan nilai pengenal VPC, tetapi paling berguna ketika digunakan sebagai variabel yang dikombinasikan dengan kunci `aws:SourceVpc` konteks. K `aws:SourceVpc` unci konteks disertakan dalam konteks permintaan hanya jika pemohon menggunakan titik akhir VPC untuk membuat permintaan. Menggunakan `aws:Ec2InstanceSourceVpc` dengan `aws:SourceVpc` memungkinkan Anda untuk menggunakan `aws:Ec2InstanceSourceVpc` lebih luas karena membandingkan nilai-nilai yang biasanya berubah bersama.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
```

```
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Dalam contoh di atas, akses ditolak jika `aws:SourceVpc` nilainya tidak sama dengan `aws:Ec2InstanceSourceVpc` nilai. Pernyataan kebijakan dibatasi hanya pada peran yang digunakan sebagai peran instans Amazon EC2 dengan menguji keberadaan kunci `ec2:SourceInstanceARN` kondisi.

Kebijakan ini digunakan `aws:ViaAWSService` untuk mengiz AWS inkan permintaan otorisasi saat permintaan dibuat atas nama peran instans Amazon EC2 Anda. Misalnya, saat Anda membuat permintaan dari instans Amazon EC2 ke bucket Amazon S3 terenkripsi, Amazon S3 melakukan AWS KMS panggilan atas nama Anda. Beberapa kunci tidak ada saat permintaan dibuat AWS KMS.

aws: Ec2InstanceSourcePrivateIPv4

Kunci ini mengidentifikasi alamat IPv4 pribadi dari antarmuka jaringan elastis utama tempat kredentif peran IAM Amazon EC2 dikirimkan. Anda harus menggunakan kunci kondisi ini dengan kunci pendampingnya `aws:Ec2InstanceSourceVpc` untuk memastikan bahwa Anda memiliki kombinasi unik global antara ID VPC dan IP pribadi sumber. Gunakan kunci ini `aws:Ec2InstanceSourceVpc` untuk memastikan bahwa permintaan dibuat dari alamat IP pribadi yang sama dengan yang dikirimkan kredenSIAL.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali pemohon menandatangani permintaan dengan kredensi peran Amazon EC2. Ini dapat digunakan dalam kebijakan IAM, kebijakan kontrol layanan, kebijakan titik akhir VPC, dan kebijakan sumber daya.
- Tipe data — alamat IP

- Jenis nilai — Single-valued

> **Important**
>
> Kunci ini tidak boleh digunakan sendiri dalam sebuah `Allow` pernyataan. Alamat IP pribadi menurut definisi tidak unik secara global. Anda harus menggunakan `aws:Ec2InstanceSourceVpc` kunci setiap kali menggunakan `aws:Ec2InstanceSourcePrivateIPv4` kunci untuk menentukan VPC dari mana kredentif instans Amazon EC2 Anda dapat digunakan.

Contoh berikut adalah kebijakan kontrol layanan (SCP) yang menolak akses ke semua sumber daya kecuali permintaan tiba melalui Titik Akhir VPC di VPC yang sama dengan kredenSIAL peran as. Dalam contoh ini, `aws:Ec2InstanceSourcePrivateIPv4` membatasi sumber kredensi ke instance tertentu berdasarkan IP sumber.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
```

```
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

aws: SourceIdentity

Gunakan kunci ini untuk membandingkan identitas sumber yang ditetapkan oleh prinsipal dengan identitas sumber yang Anda tentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setelah identitas sumber ditetapkan ketika peran diasumsikan menggunakan perintah asumsi- AWS STS role CLI, atau AWS STS `AssumeRole` operasi API.
- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci ini dalam kebijakan untuk mengizinkan tindakan AWS oleh prinsipal yang telah menetapkan identitas sumber saat mengambil peran. Aktivitas untuk identitas sumber yang ditentukan peran muncul diAWS CloudTrail. Hal ini memudahkan administrator untuk menentukan siapa atau apa yang melakukan tindakan dengan peran AWS.

Tidak seperti `sts:RoleSessionName`, setelah identitas sumber diatur, nilai tidak dapat diubah. Ini terdapat dalam konteks permintaan untuk semua tindakan yang diambil oleh peran. Nilai tetap ada dalam sesi peran berikutnya saat Anda menggunakan kredenal sesi untuk mengasumsikan peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

K `sts:SourceIdentity` unci hadir dalam permintaan ketika prinsipal awalnya menetapkan identitas sumber sambil mengambil peran menggunakan perintah asumsi- AWS STS role CLI, atau

AWS STS `AssumeRole` operasi API. Kunci `aws:SourceIdentity` tersedia dalam permintaan untuk tindakan apa pun yang diambil dengan sesi peran yang menetapkan identitas sumber.

Kebijakan kepercayaan peran berikut untuk `CriticalRole` dalam akun `111122223333` berisi kondisi untuk `aws:SourceIdentity` yang mencegah prinsipal tanpa identitas sumber yang diatur ke Saanvi atau Diego dari mengasumsikan peran.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan informasi identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

ec2: RoleDelivery

Gunakan kunci ini untuk membandingkan versi layanan metadata instans dalam permintaan yang ditandatangani dengan kredentif peran IAM untuk Amazon EC2. Layanan metadata instans membedakan antara permintaan IMDSv1 dan IMDSv2 berdasarkan apakah, untuk permintaan tertentu, baik itu header `PUT` atau, `GET` yang unik untuk IMDSv2, ada dalam permintaan itu.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali sesi peran dibuat oleh instans Amazon EC2.
- Tipe data — Numerik
- Jenis nilai — Single-valued
- Contoh nilai — 1.0, 2.0

Anda dapat mengonfigurasi Layanan Metadata Instans (IMDS) pada setiap instans yang mengharuskan kode atau pengguna lokal untuk menggunakan IMDSv2. Saat Anda menentukan bahwa IMDSv2 harus digunakan, maka IMDSv1 tidak lagi berfungsi.

- Layanan Metadata Instance Versi 1 (IMDSv1) — Sebuah metode request/response
- Layanan Metadata Instans Versi 2 (IMDSv2) - metode berorientasi sesi

Untuk informasi tentang cara mengonfigurasi instance Anda untuk menggunakan IMDSv2, lihat Meng onfigurasi opsi metadata instans.

Dalam contoh berikut, akses ditolak jika RoleDelivery nilai ec2: dalam konteks permintaan adalah 1.0 (IMDSv1). Pernyataan kebijakan ini dapat diterapkan secara umum karena, jika permintaan tidak ditandatangani oleh kredentif peran Amazon EC2, itu tidak berpengaruh.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Contoh kebijakan untuk bekerja dengan metadata instans.

ec2: SourceInstanceArn

Gunakan kunci ini untuk membandingkan ARN dari instance dari mana sesi peran dihasilkan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali sesi peran dibuat oleh instans Amazon EC2.
- Tipe data — ARN
- Jenis nilai — Single-valued
- Nilai contoh — arn:aws:ec2:us-west- 2:111111111111: -id instance/instance

Untuk contoh kebijakan, lihat Men gizinkan instance tertentu untuk melihat sumber daya di AWS layanan lain.

Iem: RoleAssumedBy

AWS Glue Layanan menetapkan kunci kondisi ini untuk setiap permintaan AWS API di mana AWS Glue membuat permintaan menggunakan peran layanan atas nama pelanggan (bukan oleh pekerjaan atau titik akhir pengembang, tetapi langsung oleh AWS Glue layanan). Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari AWS Glue layanan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat AWS Glue membuat permintaan menggunakan peran layanan atas nama pelanggan.
- Tipe data — String
- Jenis nilai — Single-valued
- Nilai contoh — Kunci ini selalu disetel `keglue.amazonaws.com`.

Contoh berikut menambahkan kondisi untuk memungkinkan AWS Glue layanan mendapatkan objek dari bucket Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
```

```
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

lem: CredentialIssuingService

AWS Glue Layanan menetapkan kunci ini untuk setiap permintaan AWS API menggunakan peran layanan yang berasal dari pekerjaan atau titik akhir pengembang. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari AWS Glue pekerjaan atau titik akhir pengembang.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat AWS Glue membuat permintaan yang berasal dari pekerjaan atau titik akhir pengembang.
- Tipe data — String
- Jenis nilai — Single-valued
- Nilai contoh — Kunci ini selalu disetel ke`glue.amazonaws.com`.

Contoh berikut menambahkan kondisi yang dilampirkan ke peran IAM yang digunakan oleh AWS Glue pekerjaan. Ini memastikan tindakan tertentu allowed/denied didasarkan pada apakah sesi peran digunakan untuk lingkungan runtime AWS Glue pekerjaan.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

pembuatan kode: BuildArn

Kunci ini mengidentifikasi CodeBuild build ARN tempat kredentif peran IAM dikirimkan. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari CodeBuild build tertentu.

> **Note**
>
> Nilai penuh tidak `codebuild:BuildArn` diketahui sebelumnya karena berisi ID build yang dihasilkan secara dinamis.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali permintaan dibuat oleh peran yang diasumsikan oleh CodeBuild.
- Tipe data — ARN
- Jenis nilai — Single-valued
- Nilai contoh — arn:aws:codebuild:us-east- 1:123456789012:: 12345678-1234-1234-1234-1234-123456789012 build/MyBuildProject

Contoh berikut memungkinkan CodeBuild build tertentu untuk memiliki `s3:GetObject` akses ke bucket yang ditentukan.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnLike": {
            "codebuild:BuildArn": "arn:aws:codebuild:us-east-1:123456789012:build/
MyBuildProject:*"
        }
    }
}
```

pembuatan kode: ProjectArn

Kunci ini mengidentifikasi CodeBuild proyek ARN yang kredenSIAL peran IAM dikirimkan ke CodeBuild build. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari CodeBuild proyek tertentu.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali permintaan dibuat oleh peran yang diasumsikan oleh CodeBuild.
- Tipe data — ARN
- Jenis nilai — Single-valued

- Nilai contoh — arn:aws:codebuild:us-east- 1:123456789012: project/MyBuildProject

Contoh berikut memungkinkan setiap build dari CodeBuild proyek tertentu untuk memiliki `s3:GetObject` akses ke bucket yang ditentukan.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnEquals": {
            "codebuild:ProjectArn": "arn:aws:codebuild:us-east-1:123456789012:project/
MyBuildProject"
        }
    }
}
```

Lambda: SourceFunctionArn

Gunakan kunci ini untuk mengidentifikasi fungsi Lambda ARN tempat kredentif peran IAM dikirimkan. Layanan Lambda menetapkan kunci ini untuk setiap permintaan AWS API yang berasal dari lingkungan eksekusi fungsi Anda. Gunakan kunci ini untuk memverifikasi apakah panggilan ke AWS sumber daya berasal dari kode fungsi Lambda tertentu. Lambda juga menetapkan kunci ini untuk beberapa permintaan yang datang dari luar lingkungan eksekusi, seperti menulis log ke CloudWatch dan mengirim jejak ke X-Ray.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali kode fungsi Lambda dipanggil.
- Tipe data — ARN
- Jenis nilai — Single-valued
- Nilai contoh — arn:aws:lambda:us-east- 1:123456789012:function: TestFunction

Contoh berikut memungkinkan satu fungsi Lambda tertentu untuk memiliki `s3:PutObject` akses bucket yang ditentukan.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Bek erja dengan kredentif lingkungan eksekusi Lambda di Panduan Peng AWS Lambda embang.

SSM: SourceInstanceArn

Gunakan kunci ini untuk mengidentifikasi ARN inst AWS Systems Manager ans terkelola tempat kredentif peran IAM dikirimkan. Kunci kondisi ini tidak ada saat permintaan berasal dari instans terkelola dengan peran IAM yang terkait dengan profil instans Amazon EC2.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan setiap kali kredentif peran dikirimkan ke instance AWS Systems Manager terkelola.
- Tipe data — ARN
- Jenis nilai — Single-valued
- Nilai contoh — arn:aws:ec2:us-west- 2:111111111111: -id instance/instance

toko identitas: UserId

Gunakan kunci ini untuk membandingkan identitas tenaga kerja Pusat Identitas IAM dalam permintaan yang ditandatangani dengan identitas yang ditentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan ketika pemanggil permintaan adalah pengguna di Pusat Identitas IAM.

- Tipe data — String
- Jenis nilai — Single-valued
- Nilai contoh — 94482488-3041-7026-18f3-be45837cd0e4

Anda dapat menemukan pengguna di Pusat Identitas IAM dengan membuat permintaan ke GetUserId API menggunakan AWS CLI, AWS API, atau AWS SDK. UserId

Properti jaringan

Gunakan kunci kondisi berikut untuk membandingkan detail tentang jaringan tempat permintaan berasal atau dilewati dengan properti jaringan yang Anda tentukan dalam kebijakan.

aws: SourceIp

Gunakan kunci ini untuk membandingkan alamat IP pemohon dengan alamat IP yang Anda tentukan dalam kebijakan. Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan, kecuali saat pemohon menggunakan titik akhir VPC untuk membuat permintaan.
- Jenis data - alamat IP
- Jenis nilai — Single-valued

Kunci kondisi `aws:SourceIp` dapat digunakan dalam kebijakan yang memungkinkan prinsipal hanya membuat permintaan dari rentang IP yang ditetapkan.

> **Note**
>
> `aws:SourceIp`mendukung alamat IPv4 dan IPv6 atau rentang alamat IP. Untuk daftar Layanan AWS dukungan IPv6, lihat Layanan AWS yang mendukung IPv6 di Panduan Pengguna Amazon VPC.

Misalnya, Anda dapat melampirkan kebijakan berbasis identitas berikut ke peran IAM. Kebijakan ini memungkinkan pengguna untuk memasukkan objek ke dalam bucket `amzn-s3-demo-bucket3` Amazon S3 jika mereka melakukan panggilan dari rentang alamat IPv4 yang ditentukan. Kebijakan ini

juga mengiz AWS inkan layanan yang menggunakan Teruskan sesi akses untuk melakukan operasi ini atas nama Anda.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

Jika Anda perlu membatasi akses dari jaringan yang mendukung pengalamatan IPv4 dan IPv6, Anda dapat menyertakan alamat IPv4 dan IPv6 atau rentang alamat IP dalam kondisi kebijakan IAM. Kebijakan berbasis identitas berikut akan memungkinkan pengguna untuk memasukkan objek ke dalam bucket `amzn-s3-demo-bucket3` Amazon S3 jika pengguna melakukan panggilan dari rentang alamat IPv4 atau IPv6 yang ditentukan. Sebelum Anda menyertakan rentang alamat IPv6 dalam kebijakan IAM Anda, verifikasi bahwa alamat yang Layanan AWS Anda gunakan mendukung IPv6. Untuk daftar Layanan AWS dukungan IPv6, lihat Layanan AWS yang mendukung IPv6 di Panduan Pengguna Amazon VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
```

```
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Jika permintaan berasal dari host yang menggunakan titik akhir VPC Amazon, maka kunci `aws:SourceIp` tidak tersedia. Sebagai gantinya, Anda harus menggunakan VPC-specific kunci seperti aws: VpcSourceIp. Untuk informasi selengkapnya tentang menggunakan titik akhir VPC, lihat Identitas dan manajemen akses untuk titik akhir VPC dan layanan titik akhir VPC di Panduan. AWS PrivateLink

> **Note**
>
> Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

aws: SourceVpc

Gunakan kunci ini untuk memeriksa apakah permintaan berjalan melalui VPC tempat titik akhir VPC dilampirkan. Dalam kebijakan, Anda dapat menggunakan kunci ini untuk memungkinkan akses hanya ke VPC tertentu. Untuk informasi selengkapnya, lihat Membatasi Akses ke VPC Tertentu di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan, hanya jika pemohon menggunakan titik akhir VPC untuk membuat permintaan.
- Tipe data — String
- Jenis nilai — Single-valued

Dalam kebijakan, Anda dapat menggunakan kunci ini untuk mengizinkan atau membatasi akses ke VPC tertentu.

Misalnya, Anda dapat melampirkan kebijakan berbasis identitas berikut ke peran IAM untuk ditolak `PutObject` ke bucket `amzn-s3-demo-bucket3` Amazon S3, kecuali permintaan dibuat dari ID VPC Layanan AWS yang ditentukan atau menggunakan sesi akses maju (FAS) untuk membuat permintaan atas nama peran tersebut. Tidak seperti denganaws: SourceIp, Anda harus menggunakan aws:ViaAWSService atau aws: CalledVia mengizinkan permintaan FAS, karena VPC sumber dari permintaan awal tidak dipertahankan.

> **Note**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Gunakan kebijakan ini bersama dengan kebijakan lain yang mengizinkan tindakan tertentu.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PutObjectIfNotVPCID",
      "Effect": "Deny",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-1234567890abcdef0"
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        }
      }
```

```
    }
  ]
}
```

> **Note**
>
> AWS merekomendasikan untuk menggunakan `aws:SourceVpcArn` alih `aws:SourceVpc` - alih jika `aws:SourceVpcArn` didukung oleh layanan yang Anda targetkan. Silakan merujuk ke aws: SourceVpcArn untuk daftar layanan yang didukung.

> **Note**
>
> Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

aws: SourceVpcArn

Gunakan kunci ini untuk memverifikasi ARN VPC yang dilewati permintaan menggunakan titik akhir VPC. Kunci ini mengembalikan ARN VPC tempat titik akhir VPC dilampirkan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan untuk layanan yang didukung saat permintaan dibuat melalui titik akhir VPC. Kunci tidak disertakan untuk permintaan yang dibuat melalui titik akhir layanan publik. Layanan berikut mendukung kunci ini:
  - AWS App Runner (awalan: `apprunner`)
  - AWS Application Discovery Service (awalan: `discovery`)
  - Amazon Athena (awalan: `athena`)
  - AWS Cloud Map (awalan: `servicediscovery`)
  - W CloudWatch awasan Aplikasi Amazon (awalan: `applicationinsights`)
  - AWS CloudFormation (awalan: `cloudformation`)

- Amazon Comprecept Medical (awalan: `comprehendmedical`)
- AWS Compute Optimizer (awalan:) `compute-optimizer`
- Amazon; Registri Kontainer Elastis (awalan: `ecr`)
- Layanan Kontainer Elastis Amazon (awalan: `ecs`)
- Amazon Kinesis Analytics (awalan: `kinesisanalytics`)
- Amazon Route 53 (awalan: `route53`)
- AWS DataSync (awalan: `datasync`)
- Toko Blok Elastis Amazon (awalan: `ebs`)
- Pen EventBridge jadwal Amazon (awalan: `scheduler`)
- Amazon Data Firehose (awalan: `firehose`)
- AWS HealthImaging (awalan: `medical-imaging`)
- AWS HealthLake (awalan: `healthlake`)
- AWS HealthOmics (awalan: `omics`)
- AWS Identity and Access Management (kecuali untuk `iam:PassRole` tindakan) (awalan: `iam`)
- AWS IoT FleetWise (awalan: `iotfleetwise`)
- AWS IoT Wireless (awalan: `iotwireless`)
- AWS Key Management Service (awalan: `kms`)
- AWS Lambda (awalan: `lambda`)
- AWS Payment Cryptography (awalan: `payment-cryptography`)
- Amazon Polly (awalan: `polly`)
- AWS Private Certificate Authority (awalan: `acm-pca`)
- AWS Recycle Bin (awalan: `rbin`)
- Amazon Rekognition (awalan:) `rekognition`
- Kuota Layanan (awalan: `servicequotas`)
- Layanan Penyimpanan Sederhana Amazon (awalan: `s3`)
- AWS Storage Gateway (awalan: `storagegateway`)
- Manajer Insiden AWS Systems Manager Kontak (awalan: `ssm-contacts`)
- Amazon Textract (awalan: `textract`)
- Transkrip Amazon (awalan: `transcribe`)
- AWS Transfer Family (awalan: `transfer`)

- Tipe data — ARN

  AWS merekomendasikan agar Anda menggunakan operator ARN al ih-alih operator string saat membandingkan ARN.

- Jenis nilai — Single-valued
- Nilai contoh — `arn:aws:ec2:us-east-1:123456789012:vpc/vpc-0e9801d129EXAMPLE`

Berikut ini adalah contoh kebijakan bucket yang menolak akses ke `amzn-s3-demo-bucket` dan objeknya dari mana saja di luar VPC`vpc-1a2b3c4d`.

```
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
       "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket",
                    "arn:aws:s3:::amzn-s3-demo-bucket/*"],
       "Condition": {
         "ArnNotEquals": {
           "aws:SourceVpcArn": "arn:aws:ec2:us-east-1:*:vpc/vpc-1a2b3c4d"
         }
       }
     }
   ]
}
```

> **Note**
>
> Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

aws: SourceVpce

Gunakan kunci ini untuk membandingkan pengidentifikasi VPC endpoint dari permintaan dengan ID titik akhir yang Anda sebutkan dalam kebijakan.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan, hanya jika pemohon menggunakan titik akhir VPC untuk membuat permintaan.
- Tipe data — String
- Jenis nilai — Single-valued

Dalam kebijakan, Anda dapat menggunakan kunci ini untuk membatasi akses ke VPC endpoint tertentu. Untuk informasi selengkapnya, lihat Membatasi akses ke VPC tertentu di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Sama seperti menggunakanaws: SourceVpc, Anda harus menggunakan aws:ViaAWSService atau mengi aws: CalledVia zinkan permintaan yang dibuat dengan Layanan AWS menggunakan sesi akses maju (FAS). Ini karena titik akhir VPC sumber dari permintaan awal tidak dipertahankan.

> **Note**
>
> Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

aws: VpceAccount

Gunakan kunci ini untuk membandingkan ID AWS akun yang memiliki titik akhir VPC tempat permintaan dibuat dengan ID akun yang Anda tentukan dalam kebijakan. Kunci kondisi ini membantu Anda menetapkan kontrol perimeter jaringan dengan memastikan permintaan datang melalui titik akhir VPC yang dimiliki oleh akun tertentu.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat permintaan dibuat melalui titik akhir VPC. Kunci tidak disertakan untuk permintaan yang dibuat melalui titik akhir layanan publik.

Layanan berikut mendukung kunci ini:

- AWS App Runner (awalan: `apprunner`)
- AWS Application Discovery Service (awalan: `discovery`)
- Amazon Athena (awalan: `athena`)
- AWS Cloud Map (awalan: `servicediscovery`)
- W CloudWatch awasan Aplikasi Amazon (awalan: `applicationinsights`)
- AWS CloudFormation (awalan: `cloudformation`)
- Amazon Comprecept Medical (awalan: `comprehendmedical`)
- AWS Compute Optimizer (awalan:) `compute-optimizer`
- Amazon; Registri Kontainer Elastis (awalan: `ecr`)
- Layanan Kontainer Elastis Amazon (awalan: `ecs`)
- Amazon Kinesis Analytics (awalan: `kinesisanalytics`)
- Amazon Route 53 (awalan: `route53`)
- AWS DataSync (awalan: `datasync`)
- Amazon DynamoDB (awalan:) `dynamodb`
- Toko Blok Elastis Amazon (awalan: `ebs`)
- Pen EventBridge jadwal Amazon (awalan: `scheduler`)
- Amazon Data Firehose (awalan: `firehose`)
- Amazon GameLift (awalan: `gamelift`)
- AWS HealthImaging (awalan: `medical-imaging`)
- AWS HealthLake (awalan: `healthlake`)
- AWS HealthOmics (awalan: `omics`)
- AWS Identity and Access Management (kecuali untuk `iam:PassRole` tindakan) (awalan: `iam`)
- AWS IoT FleetWise (awalan: `iotfleetwise`)
- AWS IoT Wireless (awalan: `iotwireless`)
- AWS Key Management Service (awalan: `kms`)
- AWS Lambda (awalan: `lambda`)
- AWS Payment Cryptography (awalan: `payment-cryptography`)
- Amazon Polly (awalan: `polly`)
- AWS Private Certificate Authority (awalan: `acm-pca`)

  - AWS Recycle Bin (awalan: `rbin`)
  - Amazon Rekognition (awalan:) `rekognition`
  - Kuota Layanan (awalan: `servicequotas`)
  - Layanan Penyimpanan Sederhana Amazon (awalan: `s3`)
  - AWS Perisai (awalan: `shield`)
  - AWS Storage Gateway (awalan: `storagegateway`)
  - Manajer Insiden AWS Systems Manager Kontak (awalan: `ssm-contacts`)
  - Amazon Textract (awalan: `textract`)
  - Transkrip Amazon (awalan: `transcribe`)
  - AWS Transfer Family (awalan: `transfer`)
- Tipe data — String
- Jenis nilai — Single-valued
- Nilai contoh — `123456789012`

Anda dapat menggunakan kunci kondisi ini untuk membatasi akses ke sumber daya sehingga permintaan harus datang melalui titik akhir VPC yang dimiliki oleh akun Anda. Contoh kebijakan bucket Amazon S3 berikut memungkinkan akses saat permintaan datang melalui titik akhir VPC yang dimiliki oleh akun tertentu:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessToSpecificVpceAccountOnly",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/RoleName"
            },
            "Action": "s3:GetObject",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
            "Condition": {
                "StringEquals": {
                    "aws:VpceAccount": "111122223333"
                }
```

```
            }
        }
    ]
}
```

> **Note**
>
> Kunci kondisi ini saat ini didukung untuk serangkaian AWS layanan tertentu. Menggunakan kunci ini dengan layanan yang tidak didukung dapat menyebabkan hasil otorisasi yang tidak diinginkan. Selalu cantumkan kunci kondisi untuk layanan yang didukung dalam kebijakan Anda.

Beberapa AWS layanan mengakses sumber daya Anda dari jaringan mereka ketika mereka bertindak atas nama Anda. Jika Anda menggunakan layanan tersebut, Anda perlu mengedit contoh kebijakan di atas untuk mengizinkan AWS layanan mengakses sumber daya Anda dari luar jaringan Anda. Untuk informasi selengkapnya tentang pola akses yang perlu diperhitungkan saat menerapkan kontrol akses berdasarkan asal permintaan, lihatTetapkan pagar pembatas izin dengan menggunakan perimeter data.

aws: VpceOrg ID

Gunakan kunci ini untuk membandingkan pengidentifikasi organisasi AWS Organizations yang memiliki titik akhir VPC tempat permintaan dibuat dengan pengidentifikasi yang Anda tentukan dalam kebijakan. Kunci kondisi ini menyediakan pendekatan yang paling dapat diskalakan untuk kontrol perimeter jaringan, secara otomatis menyertakan semua titik akhir VPC yang dimiliki oleh akun dalam organisasi Anda.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat permintaan dibuat melalui titik akhir VPC dan akun pemilik titik akhir VPC adalah anggota organisasi AWS . Kunci tidak disertakan untuk permintaan yang dibuat melalui jalur jaringan lain atau ketika akun pemilik titik akhir VPC bukan bagian dari organisasi.

  Layanan berikut mendukung kunci ini:

  - AWS App Runner (awalan: `apprunner`)
  - AWS Application Discovery Service (awalan: `discovery`)
  - Amazon Athena (awalan: `athena`)

- AWS B2B Pertukaran Data (awalan: `b2bi`)
- AWS Cloud Map (awalan: `servicediscovery`)
- AWS CloudHSM (awalan: `cloudhsmv2`)
- W CloudWatch awasan Aplikasi Amazon (awalan: `applicationinsights`)
- AWS CloudFormation (awalan: `cloudformation`)
- Amazon Cognito (awalan: `cognito-identity`)
- Amazon Compredeham Medis (awalan: `comprehendmedical`)
- AWS Compute Optimizer (awalan:) `compute-optimizer`
- AWS Database Migration Service (awalan: `dms`)
- AWS Data Layanan Direktori (awalan: `ds-data`)
- Amazon DynamoDB (awalan:) `dynamodb`
- Amazon; Registri Kontainer Elastis (awalan: `ecr`)
- Layanan Kontainer Elastis Amazon (awalan: `ecs`)
- Aliran Data Amazon Kinesis (awalan: `kinesis`)
- Amazon Kinesis Analytics (awalan: `kinesisanalytics`)
- Amazon Q Bisnis (awalan: `qbusiness`)
- Amazon Route 53 (awalan: `route53`)
- AWS DataSync (awalan: `datasync`)
- Toko Blok Elastis Amazon (awalan: `ebs`)
- Pen EventBridge jadwal Amazon (awalan: `scheduler`)
- Amazon Data Firehose (awalan: `firehose`)
- Amazon GameLift (awalan: `gamelift`)
- AWS HealthImaging (awalan: `medical-imaging`)
- AWS HealthLake (awalan: `healthlake`)
- AWS HealthOmics (awalan: `omics`)
- AWS Identity and Access Management (kecuali untuk `iam:PassRole` tindakan) (awalan: `iam`)
- AWS Toko Identitas (awalan: `identitystore`)
- AWS IoT FleetWise (awalan: `iotfleetwise`)
- AWS IoT TwinMaker (awalan: `iottwinmaker`)
- AWS IoT Wireless (awalan: `iotwireless`)

  - Ruang Kunci Amazon (untuk Apache Cassandra) (awalan:) `cassandra`
  - AWS Key Management Service (awalan: `kms`)
  - AWS Lambda (awalan: `lambda`)
  - AWS Network Firewall (awalan: `network-firewall`)
  - AWS Payment Cryptography (awalan: `payment-cryptography`)
  - Layanan SMS dan Suara Amazon Pinpoint (awalan: `sms-voice`)
  - Amazon Polly (awalan: `polly`)
  - Daftar Harga AWS (awalan: `pricing`)
  - AWS Private Certificate Authority (awalan: `acm-pca`)
  - AWS Recycle Bin (awalan: `rbin`)
  - Amazon Rekognition (awalan:) `rekognition`
  - Kuota Layanan (awalan: `servicequotas`)
  - Layanan Email Sederhana Amazon (awalan: `ses`)
  - AWS Perisai (awalan: `shield`)
  - Layanan Penyimpanan Sederhana Amazon (awalan: `s3`)
  - Layanan Antrian Sederhana Amazon (awalan: `sqs`)
  - AWS Storage Gateway (awalan: `storagegateway`)
  - Manajer Insiden AWS Systems Manager Kontak (awalan: `ssm-contacts`)
  - Amazon Textract (awalan: `textract`)
  - Transkrip Amazon (awalan: `transcribe`)
  - AWS Transfer Family (awalan: `transfer`)
  - Amazon WorkMail (awalan: `workmail`)
- Tipe data — String
- Jenis nilai — Single-valued
- Contoh nilai — `o-a1b2c3d4e5`

Contoh kebijakan kontrol sumber daya berikut menolak akses ke Amazon S3 dan AWS Key Management Service sumber daya kecuali permintaan datang melalui titik akhir VPC yang dimiliki oleh organisasi tertentu atau dari jaringan AWS layanan yang bertindak atas nama Anda. Beberapa organisasi mungkin perlu mengedit kebijakan ini lebih lanjut untuk memenuhi kebutuhan organisasi mereka, misalnya, mengizinkan akses mitra pihak ketiga. Untuk informasi selengkapnya tentang

pola akses yang perlu diperhitungkan saat menerapkan kontrol akses berdasarkan asal permintaan, lihatTetapkan pagar pembatas izin dengan menggunakan perimeter data.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceNetworkPerimeterVpceOrgID",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*",
        "kms:*"
      ],
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false",
          "aws:ViaAWSService": "false"
        },
        "StringNotEqualsIfExists": {
            "aws:VpceOrgID": "o-abcdef0123",
            "aws:PrincipalTag/network-perimeter-exception": "true"
        }
      }
    }
  ]
}
```

> **Note**
>
> Kunci kondisi ini saat ini didukung untuk serangkaian AWS layanan tertentu. Menggunakan kunci ini dengan layanan yang tidak didukung dapat menyebabkan hasil otorisasi yang tidak diinginkan. Selalu cantumkan kunci kondisi untuk layanan yang didukung dalam kebijakan Anda.

> **Note**
>
> Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

aws: VpceOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur titik akhir VPC tempat permintaan dibuat dengan jalur yang Anda tentukan dalam kebijakan. Kunci kondisi ini memungkinkan Anda untuk mengimplementasikan kontrol perimeter jaringan di tingkat unit organisasi (OU), secara otomatis menskalakan dengan penggunaan titik akhir VPC Anda saat Anda menambahkan titik akhir baru dalam OU yang ditentukan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat permintaan dibuat melalui titik akhir VPC dan akun pemilik titik akhir VPC adalah anggota organisasi. Kunci tidak disertakan untuk permintaan yang dibuat melalui jalur jaringan lain atau ketika akun pemilik titik akhir VPC bukan bagian dari organisasi.

  Layanan berikut mendukung kunci ini:

  - AWS App Runner (awalan: `apprunner`)
  - AWS Application Discovery Service (awalan: `discovery`)
  - Amazon Athena (awalan: `athena`)
  - AWS Cloud Map (awalan: `servicediscovery`)
  - W CloudWatch awasan Aplikasi Amazon (awalan: `applicationinsights`)
  - AWS CloudFormation (awalan: `cloudformation`)
  - Amazon Comprecept Medical (awalan: `comprehendmedical`)
  - AWS Compute Optimizer (awalan:) `compute-optimizer`
  - Amazon; Registri Kontainer Elastis (awalan: `ecr`)
  - Layanan Kontainer Elastis Amazon (awalan: `ecs`)

- Amazon Kinesis Analytics (awalan: `kinesisanalytics`)
- Amazon Route 53 (awalan: `route53`)
- AWS DataSync (awalan: `datasync`)
- Amazon DynamoDB (awalan:) `dynamodb`
- Toko Blok Elastis Amazon (awalan: `ebs`)
- Pen EventBridge jadwal Amazon (awalan: `scheduler`)
- Amazon Data Firehose (awalan: `firehose`)
- Amazon GameLift (awalan: `gamelift`)
- AWS HealthImaging (awalan: `medical-imaging`)
- AWS HealthLake (awalan: `healthlake`)
- AWS HealthOmics (awalan: `omics`)
- AWS Identity and Access Management (kecuali untuk `iam:PassRole` tindakan) (awalan: `iam`)
- AWS IoT FleetWise (awalan: `iotfleetwise`)
- AWS IoT Wireless (awalan: `iotwireless`)
- AWS Key Management Service (awalan: `kms`)
- AWS Lambda (awalan: `lambda`)
- AWS Payment Cryptography (awalan: `payment-cryptography`)
- Amazon Polly (awalan: `polly`)
- AWS Private Certificate Authority (awalan: `acm-pca`)
- AWS Recycle Bin (awalan: `rbin`)
- Amazon Rekognition (awalan:) `rekognition`
- Kuota Layanan (awalan: `servicequotas`)
- AWS Secrets Manager (awalan: `secretsmanager`)
- AWS Security Token Service (awalan: `sts`)
- Layanan Penyimpanan Sederhana Amazon (awalan: `s3`)
- AWS Perisai (awalan: `shield`)
- AWS Storage Gateway (awalan: `storagegateway`)
- Manajer Insiden AWS Systems Manager Kontak (awalan: `ssm-contacts`)
- Amazon Textract (awalan: `textract`)
- Transkrip Amazon (awalan: `transcribe`)

  - AWS Transfer Family (awalan: `transfer`)
- Tipe data — String (daftar)
- Jenis nilai — Multivalue
- Contoh nilai — `o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/`

Karena `aws:VpceOrgPaths` merupakan kunci kondisi multinilai, Anda harus menggunakan operator `ForAnyValue` atau `ForAllValues` set dengan operator kondisi string untuk kunci ini. Contoh kebijakan bucket Amazon S3 berikut mengizinkan akses hanya ketika permintaan datang melalui titik akhir VPC yang dimiliki oleh akun di unit organisasi tertentu:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessFromSpecificOrgPaths",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/RoleName"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:VpceOrgPaths": [
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/*",
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-33333333/*"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> Kunci kondisi ini saat ini didukung untuk serangkaian AWS layanan tertentu. Menggunakan kunci ini dengan layanan yang tidak didukung dapat menyebabkan hasil otorisasi yang tidak diinginkan. Selalu cantumkan kunci kondisi untuk layanan yang didukung dalam kebijakan Anda.

Beberapa AWS layanan mengakses sumber daya Anda dari jaringan mereka ketika mereka bertindak atas nama Anda. Jika Anda menggunakan layanan tersebut, Anda perlu mengedit contoh kebijakan di atas untuk mengizinkan AWS layanan mengakses sumber daya Anda dari luar jaringan Anda. Untuk informasi selengkapnya tentang pola akses yang perlu diperhitungkan saat menerapkan kontrol akses berdasarkan asal permintaan, lihatTetapkan pagar pembatas izin dengan menggunakan perimeter data.

aw: VpcSourceIp

Gunakan kunci ini untuk membandingkan alamat IP tempat permintaan dibuat dengan alamat IP yang Anda tentukan dalam kebijakan. Dalam kebijakan, kunci hanya cocok jika permintaan berasal dari alamat IP yang ditentukan dan melalui titik akhir VPC.

- Ketersediaan – Kunci ini disertakan dalam konteks permintaan, hanya jika permintaan dibuat menggunakan titik akhir VPC.
- Jenis data - alamat IP
- Jenis nilai — Single-valued

Untuk informasi selengkapnya, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan Pengguna Amazon VPC. Sama seperti menggunakanaws: SourceVpc, Anda harus menggunakan aws:ViaAWSService atau mengi aws: CalledVia zinkan permintaan yang dibuat dengan Layanan AWS menggunakan sesi akses maju (FAS). Ini karena IP sumber dari permintaan awal yang dibuat menggunakan titik akhir VPC tidak dipertahankan dalam permintaan FAS.

> **Note**
>
> `aws:VpcSourceIp`mendukung alamat IPv4 dan IPv6 atau rentang alamat IP. Untuk daftar Layanan AWS dukungan IPv6, lihat Layanan AWS yang mendukung IPv6 di Panduan Pengguna Amazon VPC.

Selalu gunakan tombol `aws:VpcSourceIp` kondisi dengan tombol `aws:SourceVpcaws:SourceVpce`,, atau `aws:SourceVpcArn` kondisi. Jika tidak, kebijakan mungkin mengizinkan panggilan API dari VPC tak terduga yang menggunakan CIDR IP yang sama atau tumpang tindih. Ini dapat terjadi karena IP CIDR dari dua VPC yang tidak terkait dapat sama atau tumpang tindih.

Note

Saat Layanan AWS melakukan panggilan ke orang lain Layanan AWS atas nama Anda (panggilan layanan-ke-layanan), konteks otorisasi khusus jaringan tertentu akan disunting. Jika kebijakan Anda menggunakan kunci kondisi ini dengan `Deny` pernyataan, prinsipal Layanan AWS mungkin diblokir secara tidak sengaja. Layanan AWS Agar berfungsi dengan baik sambil mempertahankan persyaratan keamanan Anda, kecualikan prinsipal layanan dari `Deny` pernyataan Anda dengan menambahkan kunci `aws:PrincipalIsAWSService` kondisi dengan nilai`false`.

Properti dari sumber daya

Gunakan kunci kondisi berikut untuk membandingkan detail tentang sumber daya yang menjadi target permintaan dengan properti sumber daya yang Anda tentukan dalam kebijakan.

aw: ResourceAccount

Gunakan kunci ini untuk membandingkan Akun AWS ID pemilik sumber daya yang diminta dengan akun sumber daya dalam kebijakan. Anda kemudian dapat mengizinkan atau menolak akses ke sumber daya tersebut berdasarkan akun yang memiliki sumber daya tersebut.

- Ketersediaan — Kunci ini selalu disertakan dalam konteks permintaan untuk sebagian besar tindakan layanan. Tindakan berikut tidak mendukung kunci ini:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`

- Amazon Elastic Block Store - Semua tindakan
- Amazon EC2
  - `ec2:AcceptTransitGatewayPeeringAttachment`
  - `ec2:AcceptVpcEndpointConnections`
  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` panggilan pada bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan acara dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Layanan Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
- AWS Security Hub CSPM
  - `securityhub:AcceptAdministratorInvitation`
- Tipe data — String
- Jenis nilai — Single-valued

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

Kunci ini sama dengan Akun AWS ID untuk akun dengan sumber daya yang dievaluasi dalam permintaan.

Untuk sebagian besar sumber daya di akun Anda, ARN berisi ID akun pemilik untuk sumber daya tersebut. Untuk sumber daya tertentu, seperti bucket Amazon S3, ARN sumber daya tidak menyertakan ID akun. Dua contoh berikut menunjukkan perbedaan antara sumber daya dengan ID akun di ARN, dan Amazon S3 ARN tanpa ID akun:

- `arn:aws:iam::123456789012:role/AWSExampleRole`— Peran IAM dibuat dan dimiliki dalam akun 123456789012.
- `arn:aws:s3:::amzn-s3-demo-bucket2`— Bucket Amazon S3 dibuat dan dimiliki dalam akun`111122223333`, tidak ditampilkan di ARN.

Gunakan AWS konsol, atau API, atau CLI, untuk menemukan semua sumber daya Anda dan ARN yang sesuai.

Anda menulis kebijakan yang menolak izin ke sumber daya berdasarkan ID akun pemilik sumber daya. Misalnya, kebijakan berbasis identitas berikut menolak akses ke sumber daya yang ditentukan jika sumber daya bukan milik akun yang ditentukan.

Untuk menggunakan kebijakan ini, ganti teks plac eholder miring dengan informasi akun Anda.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Sebaliknya, ia menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
```

```
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

Kebijakan ini menolak akses ke semua sumber daya untuk AWS layanan tertentu kecuali yang ditentukan Akun AWS memiliki sumber daya tersebut.

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang dihosting di tempat lain Akun AWS. Penggunaan `aws:ResourceAccount` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, bergantung pada akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan elemen `aws:ResourceAccount` dalam kebijakan, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil mendefinisikan pengecualian untuk sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi selengkapnya.

aw: ResourceOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur sumber daya yang diakses dengan jalur dalam kebijakan. Dalam kebijakan, kunci kondisi ini memastikan bahwa sumber daya milik anggota akun dalam root organisasi atau unit organisasi (OU) yang ditentukan di AWS

Organizations. Jal AWS Organizations ur adalah representasi teks dari struktur entitas Organisasi. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat Memahami AWS Organizations jalur entitas

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika akun yang memiliki sumber daya adalah anggota organisasi. Kunci kondisi global ini tidak mendukung tindakan berikut:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store - Semua tindakan
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` panggilan pada bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan acara dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - OpenSearch Layanan Amazon
    - `es:AcceptInboundConnection`
  - Amazon Route 53

    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- Tipe data — String (daftar)
- Jenis nilai — Multivalue

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

`aws:ResourceOrgPaths` adalah kunci kondisi multinilai. Kunci multivalue dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan operator `ForAnyValue` atau `ForAllValues` set dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Mengatur operator untuk kunci konteks multinilai.

Misalnya, kondisi berikut dikembalikan `True` untuk sumber daya milik organisasio-a1b2c3d4e5. Ketika Anda menyertakan wildcard, Anda harus menggunakan operator kondisi StringLike.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

Kondisi berikut dikembalikan `True` untuk sumber daya dengan ID OUou-ab12-11111111. Ini akan mencocokkan sumber daya yang dimiliki oleh akun yang dilampirkan ke OU ou-ab12-11111111 atau OU anak mana pun.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

Kondisi berikut dikembalikan `True` untuk sumber daya yang dimiliki oleh akun yang dilampirkan langsung ke ID OUou-ab12-22222222, tetapi bukan OU turunan. Contoh berikut menggunakan

operator StringEquals kondisi untuk menentukan persyaratan kecocokan yang tepat untuk ID OU dan bukan kecocokan wildcard.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang dihosting di tempat lain Akun AWS. Penggunaan `aws:ResourceOrgPaths` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, bergantung pada akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan `aws:ResourceOrgPaths` kunci dalam kebijakan Anda, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil mendefinisikan pengecualian untuk sumber daya milik layanan. Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam unit organisasi (OU) menggunakan `aws:ResourceOrgPaths` kunci, sambil memperhitungkan sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi selengkapnya.

aws: ResourceOrg ID

Gunakan kunci ini untuk membandingkan pengidentifikasi organisasi tempat AWS Organizations sumber daya yang diminta berada dengan pengidentifikasi yang ditentukan dalam kebijakan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika akun yang memiliki sumber daya adalah anggota organisasi. Kunci kondisi global ini tidak mendukung tindakan berikut:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`

- AWS Directory Service
  - `ds:AcceptSharedDirectory`
- Amazon Elastic Block Store - Semua tindakan
- Amazon EC2
  - `ec2:AcceptTransitGatewayPeeringAttachment`
  - `ec2:AcceptVpcEndpointConnections`
  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` panggilan pada bus acara di akun lain, jika bus acara itu dikonfigurasi sebagai EventBridge target lintas akun sebelum 2 Maret 2023. Untuk informasi selengkapnya, lihat Memberikan izin untuk mengizinkan acara dari AWS akun lain di Panduan EventBridge Pengguna Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- OpenSearch Layanan Amazon
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
- AWS Security Hub CSPM
  - `securityhub:AcceptAdministratorInvitation`
- Tipe data — String
- Jenis nilai — Single-valued

> **Note**
>
> Untuk pertimbangan tambahan untuk tindakan yang tidak didukung di atas, lihat repositori Contoh Kebijakan Perimeter Data.

Kunci global ini mengembalikan ID organisasi sumber daya untuk permintaan yang diberikan. Ini memungkinkan Anda untuk membuat aturan yang berlaku untuk semua sumber daya dalam organisasi yang ditentukan dalam `Resource` elemen kebijakan berbasis identitas. Anda dapat menentukan ID organisasi dalam elemen kondisi. Saat Anda menambahkan dan menghapus akun, kebijakan yang menyertakan `aws:ResourceOrgID` kunci secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbaruinya secara manual.

Misalnya, kebijakan berikut mencegah prinsipal menambahkan objek ke `policy-genius-dev` sumber daya kecuali sumber daya Amazon S3 milik organisasi yang sama dengan kepala sekolah yang membuat permintaan.

> **Important**
>
> Kebijakan ini tidak mengizinkan tindakan apa pun. Sebaliknya, ia menggunakan `Deny` efek yang secara eksplisit menolak akses ke semua sumber daya yang tercantum dalam pernyataan yang bukan milik akun yang terdaftar. Gunakan kebijakan ini dalam kombinasi dengan kebijakan lain yang memungkinkan akses ke sumber daya tertentu.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
        "Effect": "Deny",
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::policy-genius-dev/*",
        "Condition": {
            "StringNotEquals": {
                "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
            }
        }
```

```
    }
}
```

> **Note**
>
> Beberapa Layanan AWS memerlukan akses ke sumber daya yang AWS dimiliki yang dihosting di tempat lain Akun AWS. Penggunaan `aws:ResourceOrgID` dalam kebijakan berbasis identitas Anda dapat memengaruhi kemampuan identitas Anda untuk mengakses sumber daya ini.

AWS Layanan tertentu, seperti AWS Data Exchange, bergantung pada akses ke sumber daya di luar Anda Akun AWS untuk operasi normal. Jika Anda menggunakan `aws:ResourceOrgID` kunci dalam kebijakan Anda, sertakan pernyataan tambahan untuk membuat pengecualian untuk layanan tersebut. Kebijakan contoh AWS: Tolak akses ke sumber daya Amazon S3 di luar akun Anda kecuali AWS Data Exchange menunjukkan cara menolak akses berdasarkan akun sumber daya sambil mendefinisikan pengecualian untuk sumber daya milik layanan. Anda dapat membuat kebijakan serupa untuk membatasi akses ke sumber daya dalam organisasi menggunakan `aws:ResourceOrgID` kunci, sambil memperhitungkan sumber daya milik layanan.

Gunakan contoh kebijakan ini sebagai templat untuk membuat kebijakan kustom Anda sendiri. Lihat dokumentasi layanan Anda untuk informasi selengkapnya.

Dalam video berikut, pelajari selengkapnya tentang cara menggunakan kunci kondisi `aws:ResourceOrgID` dalam kebijakan.

Pastikan identitas dan jaringan hanya dapat digunakan untuk mengakses sumber daya tepercaya.

aw:ResourceTag/kunci tag

Gunakan kunci ini untuk membandingkan pasangan kunci-nilai tag yang Anda tentukan dalam kebijakan dengan pasangan kunci-nilai yang dilampirkan ke sumber daya. Misalnya, Anda dapat meminta agar akses ke sumber daya hanya diperbolehkan jika sumber daya memiliki kunci tanda yang dilampirkan `"Dept"` dengan nilai `"Marketing"`. Untuk informasi selengkapnya, lihat Mengontrol akses ke AWS sumber daya.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan ketika sumber daya yang diminta sudah memiliki tag terlampir atau dalam permintaan yang membuat sumber daya dengan tag

terlampir. Kunci ini hanya dikembalikan untuk sumber daya yang mendukung otorisasi berdasarkan tanda. Ada satu kunci konteks untuk setiap pasangan nilai kunci tanda.

- Tipe data — String
- Jenis nilai — Single-valued

Kunci konteks ini diformat `"aws:ResourceTag/tag-key":"tag-value"` di mana *`tag-key`* dan *`tag-value`* merupakan kunci tag dan pasangan nilai. Kunci tag tidak peka huruf besar/kecil. Ini berarti jika Anda menentukan `"aws:ResourceTag/TagKey1": "Value1"` dalam elemen ketentuan kebijakan Anda, kemudian ketentuan tersebut cocok dengan kunci tanda sumber daya bernama `TagKey1` atau `tagkey1`, tetapi tidak keduanya. Nilai dalam key/value pasangan tag ini peka huruf besar/kecil. Ini berarti bahwa jika Anda menentukan `"aws:ResourceTag/TagKey1": "Production"` dalam elemen kondisi kebijakan, kondisi tersebut cocok dengan nilai tag sumber daya yang diberi nama `Production` tetapi tidak akan cocok dengan `production` atau`PRODUCTION`.

Untuk contoh penggunaan `aws:ResourceTag` kunci untuk mengontrol akses ke sumber daya IAM, lihatMengontrol akses ke AWS sumber daya.

Untuk contoh penggunaan `aws:ResourceTag` kunci untuk mengontrol akses ke AWS sumber daya lain, lihatMengontrol akses ke AWS sumber daya menggunakan tag.

Untuk tutorial tentang menggunakan kunci kondisi `aws:ResourceTag` untuk kontrol akses berbasis atribut (ABAC), lihat Tutorial IAM: Tentukan izin untuk mengakses AWS sumber daya berdasarkan tag.

Properti permintaan

Gunakan kunci kondisi berikut untuk membandingkan detail tentang permintaan itu sendiri dan isi permintaan dengan properti permintaan yang Anda tentukan dalam kebijakan.

aws: CalledVia

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan, layanan yang membuat permintaan atas nama prinsipal IAM (pengguna atau peran). Ketika kepala sekolah membuat permintaan ke AWS layanan, layanan itu mungkin menggunakan kredenSIAL kepala sekolah untuk membuat permintaan selanjutnya ke layanan lain. Ketika permintaan dibuat menggunakan sesi akses maju (FAS), kunci ini diatur dengan nilai prinsipal layanan. Kunci `aws:CalledVia` berisi daftar yang dipesan dari setiap layanan dalam rantai yang membuat permintaan atas nama utama.

Untuk informasi selengkapnya, lihat Teruskan sesi akses.

- Ketersedian – Kunci ini tersedia dalam permintaan saat layanan yang mendukung `aws:CalledVia` menggunakan kredensial prinsipal IAM untuk mengajukan permintaan ke layanan lain. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data — String (daftar)
- Jenis nilai — Multivalue

Untuk menggunakan kunci `aws:CalledVia` kondisi dalam kebijakan, Anda harus menyediakan prinsipal layanan untuk mengizinkan atau menolak permintaan AWS layanan. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk membaca dan menulis dari tabel Amazon DynamoDB. DynamoDB kemudian menggunakan enkripsi yang disediakan oleh AWS Key Management Service ()AWS KMS.

Untuk memungkinkan atau menolak akses ketika setiap layanan membuat permintaan dengan menggunakan kredensial prinsipal, gunakan kunci kondisi `aws:ViaAWSService`. Kunci kondisi itu mendukung AWS layanan.

Kunci `aws:CalledVia` adalah tombol multinilai. Namun, Anda tidak dapat menerapkan perintah menggunakan kunci ini dalam suatu kondisi. Menggunakan contoh di atas, Pengguna 1 membuat permintaan ke CloudFormation, yang memanggil DynamoDB, yang memanggil. AWS KMS Ini adalah tiga permintaan terpisah. Panggilan terakhir AWS KMS dilakukan oleh Pengguna 1 melalui CloudFormation dan kemudian DynamoDB.
Contoh menggunakan aws:CalledVia.

Dalam kasus ini, kunci `aws:CalledVia` dalam konteks permintaan mencakup `cloudformation.amazonaws.com` dan `dynamodb.amazonaws.com`, dalam urutan tersebut. Jika Anda hanya peduli bahwa panggilan dilakukan melalui DynamoDB di suatu tempat dalam rantai permintaan, Anda dapat menggunakan kunci kondisi ini dalam kebijakan Anda.

Misalnya, kebijakan berikut memungkinkan pengelolaan AWS KMS kunci bernama`my-example-key`, tetapi hanya jika DynamoDB adalah salah satu layanan yang meminta. Operator `ForAnyValue:StringEquals` kondisi memastikan bahwa DynamoDB adalah salah satu layanan panggilan. Jika prinsipal langsung memanggil AWS KMS , kondisi akan memberikan `false` dan permintaan tersebut tidak diizinkan oleh kebijakan ini.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": [
                        "dynamodb.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Jika Anda ingin memberlakukan layanan mana yang membuat panggilan pertama atau terakhir dalam rantai, Anda dapat menggunakan kunci `aws:CalledViaFirst` dan `aws:CalledViaLast`. Misalnya, kebijakan berikut memungkinkan pengelolaan kunci yang disebutkan `my-example-key` di AWS KMS. AWS KMS Operasi ini hanya diperbolehkan jika beberapa permintaan disertakan dalam rantai. Permintaan pertama harus dilakukan melalui CloudFormation dan yang terakhir melalui DynamoDB. Jika layanan lain membuat permintaan di tengah rantai, operasi masih diizinkan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

Kunci `aws:CalledViaFirst` dan `aws:CalledViaLast` tersedia dalam permintaan saat layanan menggunakan kredensial prisipal IAM untuk menghubungi layanan lain. Mereka menunjukkan layanan pertama dan terakhir yang melakukan panggilan dalam rantai permintaan. Misalnya, asumsikan bahwa CloudFormation memanggil layanan lain bernama`X Service`, yang memanggil DynamoDB, yang kemudian memanggil. AWS KMS Panggilan terakhir AWS KMS dilakukan oleh `User 1` through CloudFormation, then`X Service`, dan kemudian DynamoDB. Ini pertama kali dipanggil melalui CloudFormation dan terakhir dipanggil melalui DynamoDB.
Contoh menggunakan aws: CalledViaFirst dan aws:CalledViaLast.

aws: CalledViaFirst

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan pertama yang mengajukan permintaan atas nama prinsipal IAM (pengguna atau peran). Untuk informasi selengkapnya, lihat `aws:CalledVia`.

- Ketersediaan – Kunci ini tersedia dalam permintaan ketika layanan menggunakan kredensial dari prinsipal IAM untuk membuat setidaknya satu permintaan lain ke layanan yang berbeda. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.

- Tipe data — String
- Jenis nilai — Single-valued

aw: CalledViaLast

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan terakhir yang mengajukan permintaan atas nama prinsipal IAM (pengguna atau peran). Untuk informasi selengkapnya, lihat `aws:CalledVia`.

- Ketersediaan – Kunci ini tersedia dalam permintaan ketika layanan menggunakan kredensial dari prinsipal IAM untuk membuat setidaknya satu permintaan lain ke layanan yang berbeda. Kunci ini tidak ada jika layanan menggunakan peran layanan atau peran terkait layanan untuk melakukan panggilan atas nama kepala sekolah. Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data — String
- Jenis nilai — Single-valued

aws:ViaAWSService

Gunakan kunci ini untuk memeriksa apakah seseorang Layanan AWS membuat permintaan ke layanan lain atas nama Anda menggunakan sesi akses maju (FAS).

Kunci konteks permintaan kembali `true` ketika layanan menggunakan sesi akses maju untuk membuat permintaan atas nama prinsipal IAM asli. Kunci konteks permintaan juga memberikan `false` saat prisipal langsung memanggil.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — Boolean
- Jenis nilai — Single-valued

aw: ViaCustomerDomain

Gunakan kunci ini untuk memeriksa apakah kepala sekolah membuat permintaan dengan menggunakan kredentif yang diperoleh dari Temporary Access Delegation (TAD). Ketika mitra atau vendor memperoleh kredenSIAL sementara melalui pertukaran token TAD, kami menetapkan kunci ini ke pengenal akun mitra yang meminta.

Gunakan kunci kondisi ini dalam kebijakan AWS Identitas dan Manajemen Akses (IAM) untuk menjangkau kontrol akses khusus ke sesi yang ditetapkan oleh delegasi akses sementara, tanpa mempengaruhi sesi Anda sendiri. Misalnya, Anda dapat menggunakan kunci ini untuk mencabut hanya sesi berbasis delegasi sambil mempertahankan akses konsol langsung.

- Ketersediaan — Kunci ini hadir dalam konteks permintaan ketika kepala sekolah menggunakan kredenSIAL yang diperoleh melalui pertukaran token TAD.
- Tipe data — String
- Jenis nilai — Single-valued

Contoh berikut menunjukkan kebijakan penolakan untuk sesi yang diperoleh melalui delegasi akses sementara dari akun mitra tertentu. Kebijakan ini hanya berlaku untuk sesi yang dikeluarkan sebelum waktu tertentu. Contoh ini menggunakan wildcard (`*`) untuk `Action` dan `Resource` dalam `Deny` pernyataan, yang merupakan pola standar untuk mencabut semua akses dari sesi tertentu. Untuk mencabut hanya tindakan tertentu, ganti wildcard dengan daftar tindakan yang sesuai.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "DateLessThan": {
                    "aws:TokenIssueTime": "2026-07-30T01:00:37.819Z"
                },
                "StringEquals": {
                    "aws:ViaCustomerDomain": "123456789012"
                }
            }
        }
    ]
}
```

aws: CalledVia AWSMCP

Gunakan kunci ini untuk membandingkan layanan dalam kebijakan dengan layanan AWS MCP yang membuat permintaan atas nama prinsipal IAM (pengguna atau peran). Ketika kepala sekolah membuat permintaan ke layanan AWS MCP, layanan tersebut menggunakan kredenSIAL

kepala sekolah untuk membuat permintaan selanjutnya ke layanan lain. Ketika permintaan dibuat menggunakan layanan AWS MCP, kunci ini diatur dengan nilai prinsipal layanan. K `aws:CalledViaAWSMCP` unci berisi nama prinsipal layanan dari layanan MCP yang membuat permintaan atas nama kepala sekolah.

- Ketersediaan — Kunci ini hadir dalam permintaan ketika layanan AWS MCP menggunakan kredenSIAL dari prinsipal IAM untuk membuat permintaan ke layanan. AWS Kunci ini juga tidak tersedia jika prinsipal menelepon langsung.
- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk mengizinkan atau menolak akses berdasarkan server MCP tertentu yang memulai permintaan. Misalnya, kebijakan berikut menolak operasi penghapusan sensitif saat diprakarsai melalui server MCP tertentu:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaSpecificMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaAWSMCP": "aws-mcp.amazonaws.com"
                }
            }
        }
    ]
}
```

AWS:melalui AWSMCPService

Gunakan kunci ini untuk memeriksa apakah layanan AWS MCP membuat permintaan ke AWS layanan lain atas nama Anda menggunakan sesi akses maju (FAS). Kunci konteks permintaan

kembali `true` ketika layanan AWS MCP meneruskan permintaan ke AWS layanan atas nama prinsipal IAM asli. Kunci konteks permintaan juga memberikan `false` saat prisipal langsung memanggil.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan ketika server AWS MCP membuat permintaan ke AWS layanan hilir atas nama prinsipal IAM.
- Tipe data — Boolean
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci ini untuk membatasi tindakan tertentu ketika mereka datang melalui server MCP. Misalnya, kebijakan berikut menolak operasi penghapusan sensitif saat dimulai melalui server AWS MCP apa pun:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "aws:ViaAWSMCPService": "true"
                }
            }
        }
    ]
}
```

aws: CurrentTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu permintaan dengan tanggal dan waktu yang Anda sebutkan dalam kebijakan. Untuk melihat contoh kebijakan yang menggunakan kunci kondisi ini, lihat AWS: Mengizinkan akses berdasarkan tanggal dan waktu.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — Tanggal
- Jenis nilai — Single-valued

aw: EpochTime

Gunakan kunci ini untuk membandingkan tanggal dan waktu permintaan dalam epoch atau waktu Unix dengan nilai yang Anda tentukan dalam kebijakan. Kunci ini juga menerima jumlah detik sejak 1 Januari 1970.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data - Tanggal, Numerik
- Jenis nilai — Single-valued

aws:referer

Gunakan kunci ini untuk membandingkan siapa yang mereferensikan permintaan di browser klien denga perujuk yang Anda tentukan dalam kebijakan. Nilai konteks permintaan `aws:referer` diberikan oleh penelepon dalam header HTTP. Header `Referer` disertakan dalam permintaan browser web saat Anda memilih tautan di halaman web. Header `Referer` berisi URL halaman web tempat tautan dipilih.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya jika permintaan ke AWS sumber daya dipanggil dengan menautkan dari URL halaman web di browser. Kunci ini tidak disertakan untuk permintaan terprogram karena tidak menggunakan tautan browser untuk mengakses sumber daya AWS .
- Tipe data — String
- Jenis nilai — Single-valued

Misalnya, Anda dapat mengakses objek Amazon S3 secara langsung menggunakan URL atau menggunakan invokasi API langsung. Untuk informasi lebih lanjut, lihat Operasi API Amazon S3 secara langsung menggunakan browser web. Saat Anda mengakses objek Amazon S3 dari URL yang ada di halaman web, URL halaman web sumber digunakan di`aws:referer`. Saat Anda mengakses objek Amazon S3 dengan mengetikkan URL ke browser Anda, tidak `aws:referer` ada. Saat Anda meminta API secara langsung, `aws:referer` juga tidak ada. Anda dapat menggunakan

kunci kondisi `aws:referer` dalam kebijakan untuk memungkinkan permintaan yang dibuat dari perujuk spesifik, seperti tautan di halaman web dalam domain perusahaan Anda.

> **Warning**
>
> Kunci ini harus digunakan dengan hati-hati. Menyertakan nilai header perujuk yang diketahui publik bukanlah sesuatu yang aman. Pihak yang tidak berwenang dapat menggunakan browser yang diubah atau disesuaikan untuk menyediakan nilai `aws:referer` yang mereka pilih. Akibatnya, tidak `aws:referer` boleh digunakan untuk mencegah pihak yang tidak berwenang membuat AWS permintaan langsung. Ini ditawarkan untuk memungkinkan pelanggan melindungi konten digital mereka, seperti konten yang disimpan di Amazon S3, agar tidak dirujuk pada pihak ketiga yang tidak berwenang.

aw: RequestedRegion

Gunakan kunci ini untuk membandingkan AWS Wilayah yang dipanggil dalam permintaan dengan Wilayah yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci kondisi global ini untuk mengontrol Wilayah mana yang dapat diminta. Untuk melihat Wil AWS ayah untuk setiap layanan, lihat T itik akhir layanan dan kuota di Referensi Umum Amazon Web Services.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — String
- Jenis nilai — Single-valued

Beberapa layanan global, seperti IAM, memiliki satu titik akhir. Karena titik akhir ini berada secara fisik di Wilayah Timur AS (N. Virginia), panggilan IAM selalu dilakukan ke Wilayah us-east-1. Misalnya, jika Anda membuat kebijakan yang menolak akses ke semua layanan jika Wilayah yang diminta bukan wilayah us-west-2, maka panggilan IAM selalu gagal. Untuk melihat contoh cara bekerja di sekitar ini, lihat NotAction dengan Penolakan.

> **Note**
>
> Kunci kondisi `aws:RequestedRegion` ini memungkinkan Anda mengontrol titik akhir layanan yang dipilih tetapi tidak mengendalikan dampak operasi. Beberapa layanan memiliki dampak lintas Wilayah..
> Misalnya, Amazon S3 memiliki operasi API yang meluas di seluruh wilayah.

- Anda dapat meminta `s3:PutBucketReplication` di satu Wilayah (yang dipengaruhi oleh kunci kondisi `aws:RequestedRegion`, tetapi Wilayah lainnya dipengaruhi berdasarkan pengaturan konfigurasi replikasi.
- Anda dapat memanggil `s3:CreateBucket` untuk membuat bucket di wilayah lain, dan menggunakan kunci `s3:LocationConstraint` kondisi untuk mengontrol wilayah yang berlaku.

Anda dapat menggunakan kunci konteks ini untuk membatasi akses ke AWS layanan dalam kumpulan Wilayah tertentu. Misalnya, kebijakan berikut memungkinkan pengguna untuk melihat semua instans Amazon EC2 di Konsol Manajemen AWS. Namun, hal ini hanya memungkinkan mereka untuk membuat perubahan pada instans di Irlandia (eu-west-1), London (eu-west-2), atau Paris (eu-west-3).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
```

```
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aw:RequestTag/kunci tag

Gunakan kunci ini untuk membandingkan pasangan nilai kunci tanda yang diteruskan dalam permintaan dengan pasangan tanda yang Anda sebutkan dalam kebijakan. Misalnya, Anda dapat memeriksa apakah permintaan tersebut menyertakan kunci tanda `"Dept"` dan memiliki nilai `"Accounting"`. Untuk informasi selengkapnya, lihat Mengontrol akses selama AWS permintaan.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan saat pasangan kunci-nilai tag diteruskan dalam permintaan. Ketika beberapa tanda diteruskan dalam permintaan, ada satu kunci konteks untuk setiap pasangan nilai kunci tanda.
- Tipe data — String
- Jenis nilai — Single-valued

Kunci konteks ini diformat `"aws:RequestTag/`*`tag-key`*`":"`*`tag-value`*`"` di mana *`tag-key`* dan *`tag-value`* merupakan kunci tag dan pasangan nilai. Kunci tag tidak peka huruf besar/kecil. Ini berarti bahwa jika Anda menentukan `"aws:RequestTag/TagKey1": "Value1"` dalam elemen kondisi kebijakan, kondisi tersebut cocok dengan kunci tag permintaan bernama salah satu `TagKey1` atau`tagkey1`, tetapi tidak keduanya. Nilai dalam key/value pasangan tag ini peka huruf besar/kecil. Ini berarti bahwa jika Anda menentukan `"aws:RequestTag/TagKey1": "Production"` dalam

elemen kondisi kebijakan, kondisi tersebut cocok dengan nilai tag permintaan yang diberi nama `Production` tetapi tidak akan cocok dengan `production` atau`PRODUCTION`.

Contoh ini menunjukkan bahwa meskipun kunci bernilai tunggal, Anda masih dapat menggunakan beberapa pasangan kunci-nilai dalam permintaan jika kuncinya berbeda.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2::111122223333:instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws: TagKeys

Gunakan kunci ini untuk membandingkan kunci tanda dalam permintaan dengan kunci yang Anda sebutkan dalam kebijakan. Sebaiknya jika Anda menggunakan kebijakan untuk mengontrol akses menggunakan tag, gunakan kunci `aws:TagKeys` kondisi untuk menentukan kunci tag apa yang diizinkan. Untuk contoh kebijakan dan informasi selengkapnya, lihat the section called “Mengontrol akses berdasarkan kunci tanda”.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan jika operasi mendukung penerusan tag dalam permintaan.
- Tipe data — String (daftar)

- Jenis nilai — Multivalue

Kunci konteks ini diformat `"aws:TagKeys":"tag-key"` di mana *`tag-key`* adalah daftar kunci tag tanpa nilai (misalnya,`["Dept","Cost-Center"]`).

Karena Anda dapat memasukkan beberapa pasangan nilai kunci tanda dalam permintaan, konten permintaan dapat berupa permintaan multinilai. Dalam hal ini, Anda harus menggunakan operator kumpulan `ForAllValues` atau `ForAnyValue`. Untuk informasi selengkapnya, lihat Mengatur operator untuk kunci konteks multinilai.

Beberapa layanan mendukung penandaan dengan operasi sumber daya, seperti membuat, mengubah, atau menghapus sumber daya. Untuk memungkinkan penandaan dan operasi sebagai satu panggilan, Anda harus membuat kebijakan yang mencakup tindakan penandaan dan tindakan pemodifikasian sumber daya. Kemudian, Anda dapat menggunakan kunci kondisi `aws:TagKeys` untuk menerapkan penggunaan kunci tanda tertentu dalam permintaan. Misalnya, untuk membatasi tag saat seseorang membuat snapshot Amazon EC2 Anda harus menyertakan `ec2:CreateSnapshot` tindakan pembuatan dan `ec2:CreateTags` menandai tindakan di dalam kebijakan. Untuk melihat kebijakan untuk skenario yang digunakan ini`aws:TagKeys`, lihat Membuat Snapshot dengan Tag di Panduan Pengguna Amazon EC2.

aws: SecureTransport

Gunakan kunci ini untuk memeriksa apakah permintaan dikirim menggunakan TLS. Konteks permintaan mengembalikan `true` atau `false`. Dalam kebijakan, Anda dapat mengizinkan tindakan tertentu hanya jika permintaan dikirim menggunakan TLS.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — Boolean
- Jenis nilai — Single-valued

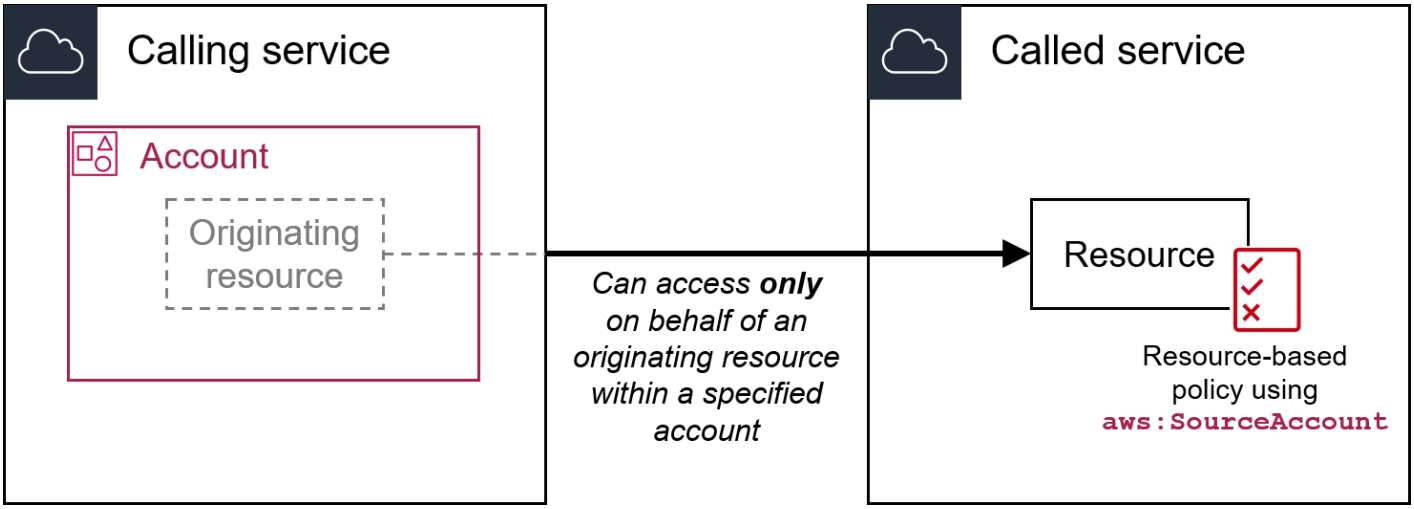
aw: SourceAccount

Gunakan kunci ini untuk membandingkan ID akun sumber daya yang membuat permintaan layanan-ke-layanan dengan ID akun yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal layanan. AWS

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan secara langsung oleh kepala sekolah AWS layanan atas nama sumber daya

yang konfigurasi memicu permintaan layanan-ke-layanan. Layanan panggilan meneruskan ID akun sumber daya asli ke layanan yang dipanggil.

> **Note**
>
> Kunci ini menyediakan mekanisme yang seragam untuk menegakkan kontrol wakil lintas layanan yang membingungkan di seluruh wilayah. Layanan AWS Namun, tidak semua integrasi layanan memerlukan penggunaan kunci kondisi global ini. Lihat dokumentasi yang Layanan AWS Anda gunakan untuk informasi lebih lanjut tentang mekanisme khusus layanan untuk mengurangi risiko wakil yang membingungkan lintas layanan.



- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk membantu memastikan bahwa layanan panggilan dapat mengakses sumber daya Anda hanya jika permintaan berasal dari akun tertentu. Misalnya, Anda dapat melampirkan kebijakan kontrol sumber daya (RCP) berikut untuk menolak permintaan oleh prinsipal layanan terhadap bucket Amazon S3, kecuali permintaan tersebut dipicu oleh sumber daya di akun yang ditentukan. Kebijakan ini menerapkan kontrol hanya pada permintaan oleh prinsipal layanan (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) yang memiliki `aws:SourceAccount` kunci present (`"Null": {"aws:SourceAccount": "false"}`), sehingga integrasi layanan yang tidak memerlukan penggunaan kunci ini dan panggilan oleh prinsipal Anda tidak terpengaruh. Jika `aws:SourceAccount` kunci hadir dalam konteks permintaan, `Null` kondisi akan dievaluasi `ketrue`, menyebabkan `aws:SourceAccount` kunci ditegakkan.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceAccount": "111122223333"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Dalam kebijakan berbasis sumber daya di mana prinsipal adalah prinsi Layanan AWS pal, gunakan kunci untuk membatasi izin yang diberikan kepada layanan. Misalnya, ketika bucket Amazon S3 dikonfigurasi untuk mengirim notifikasi ke topik Amazon SNS, layanan Amazon S3 memanggil operasi `sns:Publish` API untuk semua peristiwa yang dikonfigurasi. Dalam kebijakan topik yang memungkinkan `sns:Publish` operasi, tetapkan nilai kunci kondisi ke ID akun bucket Amazon S3.

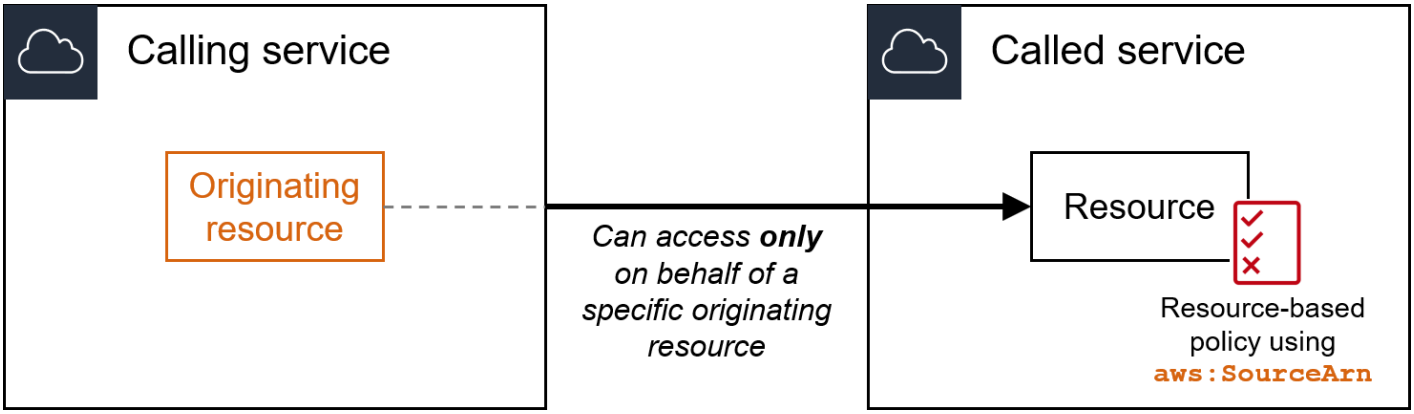
aw: SourceArn

Gunakan kunci ini untuk membandingkan Nama Sumber Daya Amazon (ARN) dari sumber daya yang membuat permintaan layanan-ke-layanan dengan ARN yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal layanan. AWS Ketika ARN sumber menyertakan ID akun, itu tidak perlu digunakan `aws:SourceAccount` dengan`aws:SourceArn`.

Kunci ini tidak bekerja dengan ARN dari prinsipal yang membuat permintaan. Sebaliknya, gunakan `aws: PrincipalArn`.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan secara langsung oleh kepala sekolah AWS layanan atas nama sumber daya yang konfigurasi memicu permintaan layanan-ke-layanan. Layanan panggilan meneruskan ARN dari sumber daya asli ke layanan yang dipanggil.

> **Note**
>
> Kunci ini menyediakan mekanisme yang seragam untuk menegakkan kontrol wakil lintas layanan yang membingungkan di seluruh wilayah. Layanan AWS Namun, tidak semua integrasi layanan memerlukan penggunaan kunci kondisi global ini. Lihat dokumentasi yang Layanan AWS Anda gunakan untuk informasi lebih lanjut tentang mekanisme khusus layanan untuk mengurangi risiko wakil yang membingungkan lintas layanan.



- Tipe data — ARN

  AWS merekomendasikan agar Anda menggunakan operator ARN al ih-alih operator string saat membandingkan ARN.

- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk membantu memastikan bahwa layanan panggilan dapat mengakses sumber daya Anda hanya jika permintaan berasal dari sumber daya tertentu. Saat menggunakan kebijakan berbasis sumber daya dengan prinsip Layanan AWS sebagai`Principal`, tetapkan nilai kunci kondisi ini ke ARN sumber daya yang ingin Anda batasi aksesnya. Misalnya, ketika bucket Amazon S3 dikonfigurasi untuk mengirim notifikasi ke topik Amazon SNS, layanan

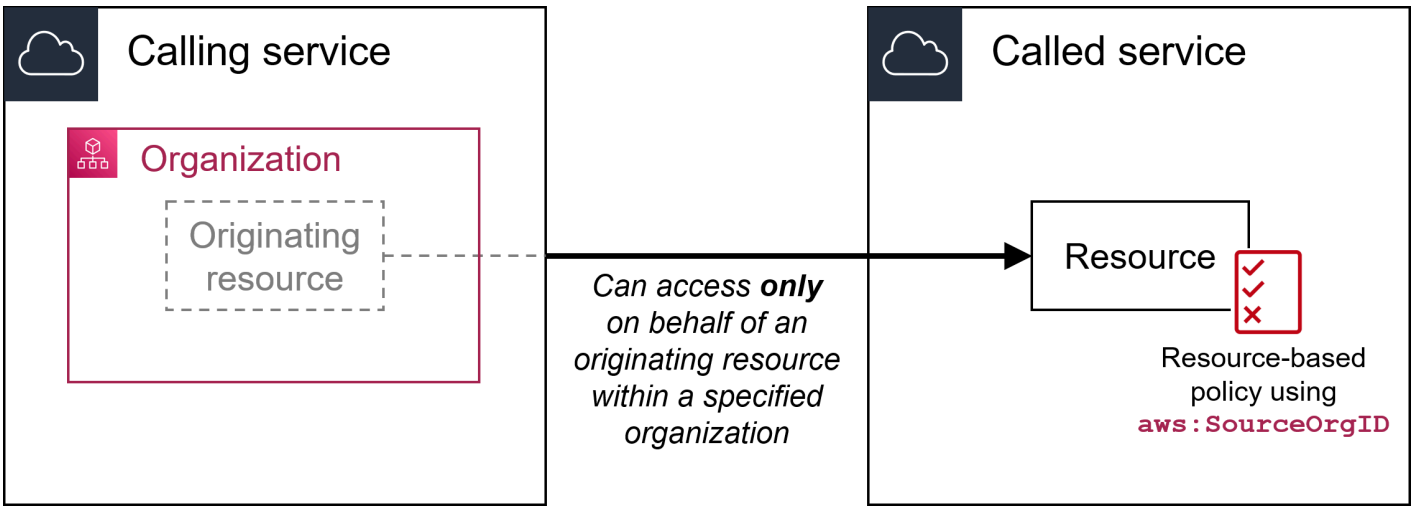
Amazon S3 memanggil operasi `sns:Publish` API untuk semua peristiwa yang dikonfigurasi. Dalam kebijakan topik yang memungkinkan `sns:Publish` operasi, tetapkan nilai kunci kondisi ke ARN bucket Amazon S3. Untuk rekomendasi tentang kapan menggunakan kunci kondisi ini dalam kebijakan berbasis sumber daya, lihat dokumentasi untuk yang Layanan AWS Anda gunakan.

aws: SourceOrg ID

Gunakan kunci ini untuk membandingkan ID https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_org_details.html organisasi sumber daya yang membuat permintaan layanan-ke-layanan dengan ID organisasi yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal layanan. AWS Saat Anda menambahkan dan menghapus akun ke organisasi di AWS Organizations, kebijakan yang menyertakan `aws:SourceOrgID` kunci secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan secara langsung oleh kepala sekolah AWS layanan atas nama sumber daya yang dimiliki oleh akun yang merupakan anggota organisasi. Layanan panggilan meneruskan ID organisasi dari sumber daya asli ke layanan yang dipanggil.

> **Note**
>
> Kunci ini menyediakan mekanisme yang seragam untuk menegakkan kontrol wakil lintas layanan yang membingungkan di seluruh wilayah. Layanan AWS Namun, tidak semua integrasi layanan memerlukan penggunaan kunci kondisi global ini. Lihat dokumentasi yang Layanan AWS Anda gunakan untuk informasi lebih lanjut tentang mekanisme khusus layanan untuk mengurangi risiko wakil yang membingungkan lintas layanan.



- Tipe data — String
- Jenis nilai — Single-valued

Anda dapat menggunakan kunci kondisi ini untuk membantu memastikan bahwa layanan panggilan dapat mengakses sumber daya Anda hanya jika permintaan berasal dari organisasi tertentu. Misalnya, Anda dapat melampirkan kebijakan kontrol sumber daya (RCP) berikut untuk menolak permintaan oleh prinsipal layanan terhadap bucket Amazon S3, kecuali permintaan tersebut dipicu oleh sumber daya di organisasi yang ditentukan AWS . Kebijakan ini menerapkan kontrol hanya pada permintaan oleh prinsipal layanan (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) yang memiliki `aws:SourceAccount` kunci present (`"Null": {"aws:SourceAccount": "false"}`), sehingga integrasi layanan yang tidak memerlukan penggunaan kunci dan panggilan oleh prinsipal Anda tidak terpengaruh. Jika `aws:SourceAccount` kunci hadir dalam konteks permintaan, `Null` kondisi akan dievaluasi `ketrue`, menyebabkan `aws:SourceOrgID` kunci ditegakkan. Kami menggunakan `aws:SourceOrgID` al `aws:SourceAccount` ih-alih operator `Null` kondisi sehingga kontrol tetap berlaku jika permintaan berasal dari akun yang bukan milik organisasi.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
```

```
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-xxxxxxxxxx"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

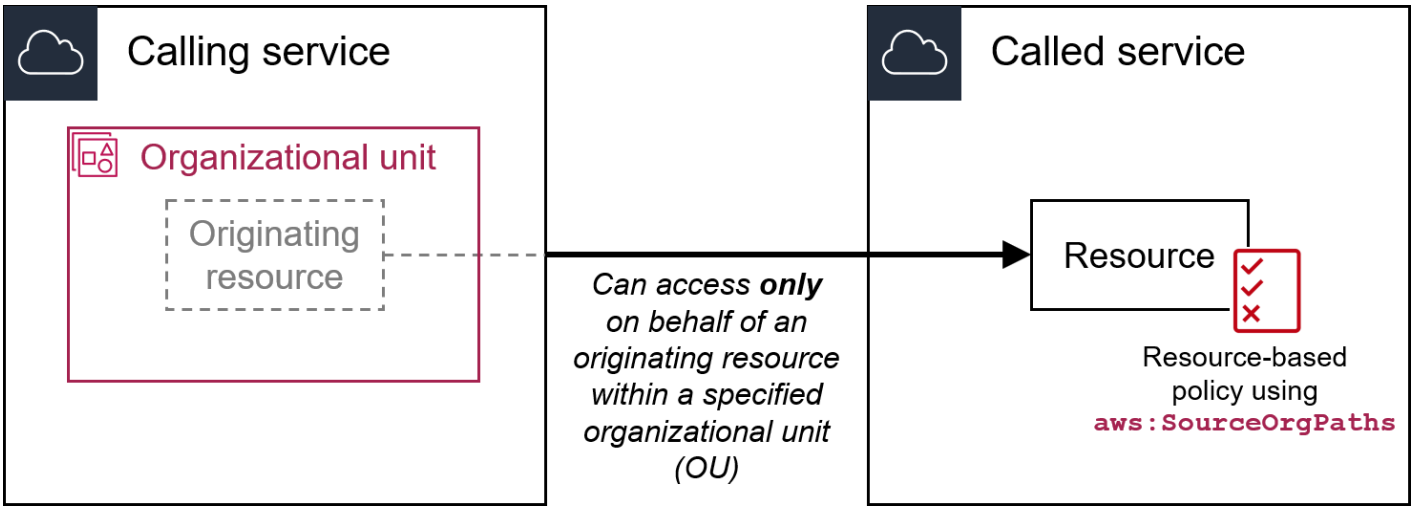
aw: SourceOrgPaths

Gunakan kunci ini untuk membandingkan AWS Organizations jalur sumber daya yang membuat permintaan layanan-ke-layanan dengan jalur organisasi yang Anda tentukan dalam kebijakan, tetapi hanya jika permintaan dibuat oleh prinsipal layanan. AWS Jal AWS Organizations ur adalah representasi teks dari struktur AWS Organizations entitas. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat Memahami jalur AWS Organizations entitas.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan hanya ketika panggilan ke sumber daya Anda dilakukan secara langsung oleh kepala sekolah AWS layanan atas nama sumber daya yang dimiliki oleh akun yang merupakan anggota organisasi. Layanan panggilan melewati jalur organisasi dari sumber daya asli ke layanan yang dipanggil.

> **Note**
>
> Kunci ini menyediakan mekanisme yang seragam untuk menegakkan kontrol wakil lintas layanan yang membingungkan di seluruh wilayah. Layanan AWS Namun, tidak semua integrasi layanan memerlukan penggunaan kunci kondisi global ini. Lihat dokumentasi

yang Layanan AWS Anda gunakan untuk informasi lebih lanjut tentang mekanisme khusus layanan untuk mengurangi risiko wakil yang membingungkan lintas layanan.



- Tipe data — String (daftar)
- Jenis nilai — Multivalue

Gunakan kunci kondisi ini untuk membantu memastikan bahwa layanan panggilan dapat mengakses sumber daya Anda hanya jika permintaan berasal dari unit organisasi tertentu (OU) di AWS Organizations.

Demikian pula`aws:SourceOrgID`, untuk membantu mencegah dampak pada integrasi layanan yang tidak memerlukan penggunaan kunci ini, gunakan operator `Null` `aws:SourceAccount` kondisi dengan kunci kondisi sehingga kontrol tetap berlaku jika permintaan berasal dari akun yang bukan milik organisasi.

```
{
      "Condition": {
        "ForAllValues:StringNotLikeIfExists": {
            "aws:SourceOrgPaths": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-
ab12-22222222/"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
```

```
        }
      }
}
```

`aws:SourceOrgPaths` adalah kunci kondisi multinilai. Kunci multivalue dapat memiliki beberapa nilai dalam konteks permintaan. Anda harus menggunakan operator `ForAnyValue` atau `ForAllValues` set dengan operator kondisi string untuk kunci ini. Untuk informasi selengkapnya tentang kunci kondisi multinilai ini, lihat Mengatur operator untuk kunci konteks multinilai.

aws: UserAgent

Gunakan kunci ini untuk membandingkan aplikasi klien pemohon dan aplikasi yang Anda tentukan dalam kebijakan.

- Ketersediaan – Kunci ini selalu disertakan dalam konteks permintaan.
- Tipe data — String
- Jenis nilai — Single-valued

> **Warning**
>
> Kunci ini harus digunakan dengan hati-hati. Sejak `aws:UserAgent` nilai disediakan oleh pemanggil dalam header HTTP, pihak yang tidak berwenang dapat menggunakan browser yang diubah atau disesuaikan untuk menyediakan nilai `aws:UserAgent` yang mereka pilih. Akibatnya, tidak `aws:UserAgent` boleh digunakan untuk mencegah pihak yang tidak berwenang membuat AWS permintaan langsung. Anda dapat menggunakannya hanya untuk mengizinkan aplikasi klien tertentu, dan hanya setelah menguji kebijakan Anda.

aw: IsMcpServiceAction

Gunakan kunci ini untuk memverifikasi bahwa tindakan yang diotorisasi adalah tindakan Layanan MCP. Kunci ini tidak mengacu pada tindakan yang diambil oleh layanan MCP ke AWS layanan lain.

- Ketersediaan — Kunci ini disertakan dalam konteks permintaan dan disetel ke True hanya ketika layanan MCP mengotorisasi tindakan layanan MCP.
- Tipe data — Boolean
- Jenis nilai — Single-valued

Kunci kondisi lintas layanan lainnya

AWS STS mendukung kunci kondisi SAML-based federasi dan kunci kondisi lintas layanan untuk federasi OIDC. Kunci ini tersedia ketika pengguna yang digabungkan menggunakan OIDC atau SAML melakukan AWS operasi di layanan lain.

IAM dan AWS STS kunci konteks kondisi

Anda dapat menggunakan `Condition` elemen dalam kebijakan JSON untuk menguji nilai kunci yang disertakan dalam konteks permintaan semua AWS permintaan. Kunci-kunci ini menyediakan informasi tentang permintaannya sendiri atau sumber daya yang dirujuk oleh permintaan. Anda dapat memeriksa bahwa kunci telah menentukan nilai sebelum mengizinkan tindakan yang diminta oleh pengguna. Hal ini memberi Anda kendali terperinci ketika pernyataan kebijakan JSON Anda cocok atau tidak cocok dengan permintaan masuk. Untuk informasi tentang penggunaan elemen `Condition` dalam kebijakan JSON, lihat Elemen kebijakan IAM JSON: Kondisi.

Topik ini menjelaskan kunci yang ditentukan dan disediakan oleh layanan IAM (dengan `iam:` awalan) dan layanan AWS Security Token Service (AWS STS) (dengan `sts:` awalan). Beberapa AWS layanan lain juga menyediakan kunci khusus layanan yang relevan dengan tindakan dan sumber daya yang ditentukan oleh layanan tersebut. Untuk informasi selengkapnya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS Layanan. Dokumentasi untuk layanan yang mendukung kunci kondisi sering kali memiliki informasi tambahan. Misalnya, untuk informasi tentang kunci yang dapat Anda gunakan dalam kebijakan untuk sumber daya Amazon S3, lihat Kunci Kebijakan Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Topik

- Kunci yang tersedia untuk IAM
- Kunci yang tersedia untuk AWS Federasi OIDC
- Kunci yang tersedia untuk SAML-based AWS STS federasi
- Cross-service SAML-based AWS STS kunci konteks federasi
- Kunci yang tersedia untuk AWS STS

Kunci yang tersedia untuk IAM

Anda dapat menggunakan kunci-kunci kondisi berikut dalam kebijakan yang mengontrol akses ke sumber daya IAM:

sudah: AccountPropertyNamespaces

Bekerja dengan operator string.

Menentukan ruang nama properti akun yang dibaca atau dimodifikasi dalam permintaan. Properti akun diatur ke dalam ruang nama (misalnya,`RoleManager`). Saat Anda memanggil `PutAccountProperties` atau`GetAccountProperties`, gunakan kunci kondisi ini untuk membatasi ruang nama properti mana yang dapat diakses oleh kepala sekolah.

sudah: AssociatedResourceArn

Bekerja IT Operator ARN .

Menentukan ARN sumber daya yang akan dihubungkan dengan peran ini di layanan tujuan. Sumber daya biasanya dimiliki oleh layanan yang perannya diteruskan oleh prinsipal. Terkadang, sumber daya mungkin dimiliki oleh layanan ketiga. Sebagai contoh, Anda mungkin meneruskan peran ke Amazon EC2 Auto Scaling yang mereka gunakan di instans Amazon EC2. Dalam kasus ini, kondisi akan cocok dengan ARN dari instans Amazon EC2.

Kunci kondisi ini hanya berlaku untuk tindakan PassRole di dalam kebijakan. Kondisi ini tidak dapat digunakan untuk membatasi tindakan lainnya.

> **Important**
>
> Saat menggunakan `iam:AssociatedResourceArn` kondisi dalam kebijakan untuk membatasi PassRole tindakan, pertimbangan khusus berlaku jika kebijakan dimaksudkan untuk menentukan akses untuk AddRoleToInstanceProfile tindakan tersebut. Dalam hal ini, Anda tidak dapat menentukan Region atau ID instans di ARN instans EC2. Nilai ARN harus `arn:aws:ec2:*:CallerAccountId:instance/*` Menggunakan nilai ARN lainnya dapat menyebabkan hasil evaluasi yang tidak terduga.

Gunakan kunci kondisi ini dalam kebijakan berbasis identitas untuk mengizinkan entitas meneruskan peran, tetapi hanya jika peran tersebut dikaitkan dengan sumber daya yang ditentukan. Misalnya, Anda dapat mengizinkan pengguna atau peran IAM untuk meneruskan peran apa pun ke layanan Amazon EC2 untuk digunakan dengan instans di. Akun AWS Pengguna atau peran IAM tidak akan diizinkan untuk meneruskan peran ke layanan lain.

```
{
```

```
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "ec2.amazonaws.com"
        },
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:*:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS layanan yang mendukung iam: PassedToService juga mendukung kunci kondisi ini.

sudah: AWSServiceName

Bekerja dengan operator string.

Menentukan AWS layanan yang dilampirkan peran ini.

Kunci kondisi ini didukung oleh operasi `CreateServiceLinkedRole` API.

> **Tip**
>
> Untuk informasi tentang layanan mana yang mendukung penggunaan peran terkait layanan, lihat AWS layanan yang bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Service-Linked Per an. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Dalam contoh ini, Anda mengizinkan entitas untuk membuat peran terkait layanan menggunakan operasi `CreateServiceLinkedRole` API jika nama layanannya access-analyzer.amazonaws.com.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
     }]
 }
```

sudah: FIDO-certification

Bekerja dengan operator string.

Memeriksa tingkat sertifikasi FIDO perangkat MFA pada saat pendaftaran kunci keamanan FIDO. Sertifikasi perangkat diambil dari FIDO Alliance Metadata Service (MDS). Jika status sertifikasi atau tingkat kunci keamanan FIDO Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L1plus, L2, L2plus, L3, L3plus

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIDO Level 1 plus untuk perangkat Anda.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

sudah: FIDO-FIPS-140-2-certification

Bekerja dengan operator string.

Memeriksa tingkat sertifikasi FIPS-140-2 validasi perangkat MFA pada saat pendaftaran kunci keamanan FIDO. Sertifikasi perangkat diambil dari FIDO Alliance Metadata Service (MDS). Jika status sertifikasi atau tingkat kunci keamanan FIDO Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L2, L3, L4

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS-140-2 Level 2 untuk perangkat Anda.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

sudah: FIDO-FIPS-140-3-certification

Bekerja dengan operator string.

Memeriksa tingkat sertifikasi FIPS-140-3 validasi perangkat MFA pada saat pendaftaran kunci keamanan FIDO. Sertifikasi perangkat diambil dari FIDO Alliance Metadata Service (MDS). Jika status sertifikasi atau tingkat kunci keamanan FIDO Anda berubah, itu tidak akan diperbarui kecuali perangkat tidak terdaftar dan terdaftar lagi untuk mengambil informasi sertifikasi yang diperbarui.

Nilai yang mungkin dari L1, L2, L3, L4

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS-140-3 Level 3 untuk perangkat Anda.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

}
```

sudah: OrganizationsPolicyId

Bekerja dengan operator string.

Memeriksa apakah kebijakan dengan AWS Organizations ID yang ditentukan cocok dengan kebijakan yang digunakan dalam permintaan. Untuk melihat contoh kebijakan IAM yang menggunakan kunci kondisi ini, lihat IAM: Lihat informasi layanan yang terakhir diakses untuk kebijakan AWS Organizations.

sudah: PassedToService

Bekerja dengan operator string.

Menentukan prinsipal layanan untuk peran yang dapat diteruskan. Kunci kondisi ini hanya berlaku untuk tindakan PassRole di dalam kebijakan. Kondisi ini tidak dapat digunakan untuk membatasi tindakan lainnya.

Saat Anda menggunakan kunci kondisi ini dalam suatu kebijakan, tentukan layanan menggunakan prinsipal layanan. Prinsipal layanan adalah nama layanan yang dapat ditentukan dalam elemen `Principal` dari suatu kebijakan. Ini adalah format biasa: `SERVICE_NAME_URL.amazonaws.com`.

Anda dapat menggunakan `iam:PassedToService` untuk membatasi pengguna Anda sehingga mereka dapat meneruskan peran ke layanan tertentu. Misalnya, pengguna dapat membuat peran layanan yang mempercayai CloudWatch untuk menulis data log ke bucket Amazon S3

atas nama mereka. Kemudian pengguna harus memberlakukan kebijakan izin dan kebijakan kepercayaan pada peran layanan baru. Dalam hal ini, kebijakan kepercayaan harus menyebutkan `cloudwatch.amazonaws.com` dalam elemen `Principal`. Untuk melihat kebijakan yang memungkinkan pengguna meneruskan peran CloudWatch, lihatIAM: Lulus peran IAM ke layanan tertentu AWS.

Dengan menggunakan kunci kondisi ini, Anda dapat memastikan bahwa pengguna membuat peran layanan hanya untuk layanan yang Anda tentukan. Misalnya, jika pengguna dengan kebijakan sebelumnya mencoba membuat peran layanan untuk Amazon EC2, operasi akan gagal. Kegagalan terjadi karena pengguna tidak memiliki izin untuk meneruskan peran ke Amazon EC2.

Kadang-kadang Anda meneruskan peran ke layanan yang kemudian memberikan peran ke layanan lain. `iam:PassedToService` hanya mencakup layanan akhir yang mengasumsikan peran, bukan layanan perantara yang meneruskan peran.

> **Note**
>
> Beberapa layanan tidak mendukung kunci kondisi ini.

sudah: PermissionsBoundary

Bekerja IT Operator ARN .

Memastikan bahwa kebijakan yang ditentukan diberlakukan sebagai batas izin pada sumber daya utama IAM. Untuk informasi selengkapnya, lihat Batas izin untuk entitas IAM

iam:PolicyARN

Bekerja IT Operator ARN .

Memeriksa Amazon Resource Name (ARN) dari kebijakan terkelola dengan permintaan yang melibatkan kebijakan terkelola. Untuk informasi selengkapnya, lihat Mengendalikan akses ke kebijakan.

Nama: ARN RoleTemplate

Bekerja IT Operator ARN .

Menentukan ARN template peran yang digunakan untuk membuat atau mengelola peran. Ketika AWS layanan memperoleh peran dari template peran, template peran ARN

disertakan sebagai kunci konteks dalam pemeriksaan otorisasi untuk`CreateRole`,, `TagRole``PutRolePolicy`,`AttachRolePolicy`,`PutRolePermissionsBoundary`, dan`GetRole`.

sudah: RegisterSecurityKey

Bekerja dengan operator string.

Memeriksa status pengaktifan perangkat MFA saat ini.

Nilai yang mungkin dari `Create` atau`Activate`.

Dalam contoh ini, Anda mendaftarkan kunci keamanan dan mengambil sertifikasi FIPS-140-3 Level 1 untuk perangkat Anda.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

 }
```

Sudah:/ResourceTag*key-name*

Bekerja dengan operator string.

Memeriksa apakah tanda yang dilampirkan ke sumber daya identitas (pengguna atau peran) cocok dengan nama dan nilai kunci yang ditentukan.

> **Note**
>
> IAM dan AWS STS mendukung kunci kondisi `iam:ResourceTag` IAM dan kunci kondisi `aws:ResourceTag` global.

Anda dapat menambahkan atribut khusus ke sumber daya IAM dalam bentuk pasangan kunci-nilai. Untuk informasi selengkapnya tentang tag untuk sumber daya IAM, lihatthe section called "Tag untuk sumber daya IAM". Anda dapat menggunakan `ResourceTag` untuk mengontrol akses ke AWS sumber daya, termasuk sumber daya IAM. Namun, karena IAM tidak mendukung tag untuk grup, Anda tidak dapat menggunakan tag untuk mengontrol akses ke grup.

Contoh ini menunjukkan bagaimana Anda dapat membuat kebijakan berbasis identitas yang memungkinkan menghapus pengguna dengan tag. **`status=terminated`** Untuk menggunakan kebijakan ini, ganti kebijakan *`italicized placeholder text`* dalam contoh dengan informasi Anda sendiri. Lalu, ikuti petunjuk di buat kebijakan atau edit kebijakan.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status":
 "terminated"}}
    }]
}
```

sudah: ServiceSpecificCredentialAgeDays

Bekerja dengan operator numerik.

Kunci kondisi ini membatasi pembuatan kredenSIAL khusus layanan berdasarkan pengaturan kedaluwarsanya. Ini memungkinkan Anda untuk mengontrol usia maksimum, dalam hari, dari kredenSIAL khusus layanan yang dapat dibuat.

Rentang yang berlaku untuk hari adalah 1 hingga 36600 (minimal 1 hari, maksimum 36600 hari).

Kunci kondisi ini didukung oleh operasi `CreateServiceSpecificCredential` API.

Dalam contoh ini, Anda mengizinkan pengguna membuat kredenSIAL khusus layanan untuk layanan Amazon Bedrock hanya jika kredensialnya kedaluwarsa dalam 90 hari.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                },
                "NumericLessThanEquals": {
                    "iam:ServiceSpecificCredentialAgeDays": "90"
                }
            }
        }
    ]
}
```

sudah: ServiceSpecificCredentialServiceName

Bekerja dengan operator string.

Menentukan AWS layanan mana yang dapat digunakan saat mengelola kredenSIAL khusus layanan. Kunci kondisi ini memungkinkan Anda membatasi AWS layanan mana yang diizinkan saat mengelola kredenSIAL khusus layanan.

Kunci kondisi ini didukung oleh operasi API berikut:

- `CreateServiceSpecificCredential`

- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential
- UpdateServiceSpecificCredential

Layanan berikut didukung untuk kredenSIAL khusus layanan dengan pemformatan nilai yang tepat:

- `bedrock.amazonaws.com`
- `cassandra.amazonaws.com`
- `codecommit.amazonaws.com`

Dalam contoh ini, Anda mengizinkan pengguna membuat kredenSIAL khusus layanan menggunakan operasi CreateServiceSpecificCredential API hanya untuk layanan Amazon Bedrock.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                }
            }
        }
    ]
}
```

sudah: DelegationDuration

Bekerja dengan operator numerik.

Memfilter akses berdasarkan durasi akses sementara yang diminta dalam permintaan delegasi.

Penyedia produk dapat menggunakan kunci kondisi ini untuk mengontrol durasi maksimum yang mereka izinkan dalam permintaan delegasi yang dikirim ke pelanggan. Durasi ditentukan

dalam detik dan menentukan berapa lama kredenSIAL sementara tetap valid setelah pelanggan melepaskan token pertukaran. Ini membantu penyedia produk menegakkan kebijakan internal seputar batas durasi akses berdasarkan kasus penggunaan mereka.

Kunci kondisi ini didukung oleh operasi `CreateDelegationRequest` API.

Dalam contoh ini, Anda mengizinkan pembuatan permintaan delegasi hanya jika durasi yang diminta adalah 7200 detik (2 jam) atau kurang.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "NumericLessThanEquals": {
                    "iam:DelegationDuration": "7200"
                }
            }
        }
    ]
}
```

sudah: NotificationChannel

Bekerja IT Operator ARN .

Memfilter akses berdasarkan topik Amazon SNS ARN yang ditentukan untuk menerima pemberitahuan permintaan delegasi.

Penyedia produk dapat menggunakan kunci kondisi ini untuk membatasi topik SNS mana yang dapat digunakan untuk pemberitahuan permintaan delegasi dalam panggilan CreateDelegationRequest API. Penyedia produk harus menentukan topik SNS untuk menerima pemberitahuan perubahan status dan token pertukaran. Ini memastikan bahwa notifikasi dikirim hanya ke saluran yang disetujui dalam organisasi penyedia produk.

Kunci kondisi ini didukung oleh operasi `CreateDelegationRequest` API.

Dalam contoh ini, Anda mengizinkan pembuatan permintaan delegasi hanya jika mereka menggunakan topik SNS tertentu untuk pemberitahuan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:NotificationChannel": "arn:aws:sns:us-
east-1:123456789012:delegation-notifications"
                }
            }
        }
    ]
}
```

sudah: TemplateArn

Bekerja IT Operator ARN .

Memfilter akses berdasarkan template kebijakan ARN yang digunakan untuk menentukan izin dalam permintaan delegasi.

Penyedia produk dapat menggunakan kunci kondisi ini untuk mengontrol templat kebijakan mana yang dapat digunakan dalam panggilan CreateDelegationRequest API. Templat kebijakan menentukan izin sementara yang diminta penyedia produk di akun pelanggan. Hal ini memungkinkan penyedia produk membatasi templat kebijakan terdaftar mana yang dapat digunakan saat membuat permintaan delegasi.

Kunci kondisi ini didukung oleh operasi `CreateDelegationRequest` API.

Dalam contoh ini, Anda mengizinkan pembuatan permintaan delegasi hanya jika mereka menggunakan templat kebijakan dari domain mitra tertentu.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
```

```
            "Condition": {
                "ArnLike": {
                    "iam:TemplateArn": "arn:aws:iam::*:delegation-template/
partner_*"
                }
            }
        }
    ]
}
```

sudah: DelegationRequestOwner

Bekerja IT Operator ARN .

Memfilter akses berdasarkan AWS identitas atau prinsipal yang memiliki permintaan delegasi.

Pelanggan dapat menggunakan kunci kondisi ini untuk mengontrol siapa yang dapat melakukan tindakan pada permintaan delegasi berdasarkan kepemilikan. Pemilik permintaan delegasi adalah AWS identitas atau prinsipal di akun pelanggan yang memulai atau menerima permintaan delegasi.

Kunci kondisi ini didukung oleh operasi API berikut:

- `GetDelegationRequest`
- `AcceptDelegationRequest`
- `RejectDelegationRequest`
- `SendDelegationToken`
- `ListDelegationRequests`
- `UpdateDelegationRequest`

Dalam contoh ini, Anda mengizinkan pengguna untuk mengelola hanya permintaan delegasi yang mereka miliki.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetDelegationRequest",
                "iam:AcceptDelegationRequest",
```

```
                "iam:RejectDelegationRequest",
                "iam:SendDelegationToken",
                "iam:UpdateDelegationRequest",
                "iam:ListDelegationRequests"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:DelegationRequestOwner": "${aws:PrincipalArn}"
                }
            }
        }
    ]
}
```

Kunci yang tersedia untuk AWS Federasi OIDC

Anda dapat menggunakan federasi OIDC untuk memberikan kredentif keamanan sementara kepada pengguna yang telah diautentikasi melalui penyedia identitas yang kompatibel (IdP) OpenID Connect ke penyedia identitas IAM OpenID Connect (OIDC) di akun Anda. AWS Contoh penyedia tersebut termasuk GitHub, Amazon Cognito, Login dengan Amazon, dan Google. Token identitas dan token akses dari IdP Anda sendiri dapat digunakan, serta token akun layanan yang diberikan ke beban kerja Amazon Elastic Kubernetes Service.

Anda dapat menggunakan kunci konteks kondisi AWS OIDC untuk menulis kebijakan yang membatasi akses prinsipal federasi ke sumber daya yang terkait dengan penyedia, aplikasi, atau pengguna tertentu. Kunci ini biasanya digunakan dalam kebijakan kepercayaan untuk suatu peran. Tentukan kunci kondisi menggunakan nama penyedia OIDC (`token.actions.githubusercontent.com`) diikuti dengan klaim (`:aud`):**`token.actions.githubusercontent.com:aud`**.

Beberapa kunci kondisi federasi OIDC dapat digunakan dalam sesi peran untuk mengotorisasi akses sumber daya. Jika nilainya Ya di kolom Ter sedia dalam sesi, Anda dapat menggunakan kunci kondisi ini dalam kebijakan untuk menentukan pengguna apa yang diizinkan untuk mengakses AWS layanan lain. Bila klaim tidak tersedia dalam sesi, kunci konteks kondisi OIDC hanya dapat digunakan dalam kebijakan kepercayaan peran untuk AssumeRoleWithWebIdentity otentikasi awal.

Pilih IdP Anda untuk melihat bagaimana klaim dari IdP Anda dipetakan ke kunci konteks kondisi IAM masuk. AWS Informasi lebih lanjut tentang kunci untuk GitHub dan Google dapat ditemukan di bawah tab Default.

Default

Default mencantumkan klaim OIDC standar dan bagaimana mereka memetakan ke kunci konteks AWS STS kondisi di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP. Gunakan pemetaan ini jika IdP Anda tidak tercantum dalam opsi tab.

GitHub Alur kerja tindakan dan Google adalah beberapa contoh IdPs yang menggunakan implementasi default di token ID JWT OIDC mereka.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
amr	amr	Ya
aud	azp Jika tidak ada nilai yang ditetapkan untuk`azp`, kunci `aud` kondisi dipetakan ke `aud` klaim.	Ya
Email	Email	Tidak
oaud	aud	Tidak
sub	sub	Ya

Untuk informasi selengkapnya tentang menggunakan kunci konteks kondisi OIDC dengan GitHub, lihatMengkonfigurasi peran untuk penyedia identitas GitHub OIDC. Untuk informasi selengkapnya tentang kolom `aud` dan `azp` Google, lihat Panduan Google Identity Platform OpenID Connect.

amr

Bekerja dengan operator string. Kunci ini memiliki banyak nilai, artinya jika Anda mengujinya dalam suatu kebijakan yang menggunakan operator kumpulan kondisi.

Contoh: `token.actions.githubusercontent.com:amr`

Referensi Metode Otentikasi mencakup informasi login tentang pengguna. Kunci dapat berisi nilai-nilai berikut:

- Jika pengguna tidak autentikasi, kunci hanya berisi `unauthenticated`.
- Jika pengguna diautentikasi, kunci berisi nilai `authenticated` dan nama penyedia login yang digunakan dalam panggilan (`accounts.google.com`).

Aud

Bekerja dengan operator string.

Contoh:

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Gunakan kunci `aud` kondisi untuk memverifikasi bahwa audiens cocok dengan yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci aud dengan sub kunci untuk penyedia identitas yang sama.

Kunci kondisi ini disetel dari bidang token berikut:

- `aud` untuk ID klien Google OAuth 2.0 dari aplikasi Anda, jika kolom `azp` belum ditetapkan. Ketika `azp` bidang disetel, `aud` bidang tersebut cocok dengan kunci `accounts.google.com:oaud` kondisi.
- `azp` jika kolom `azp` ditetapkan. Ini dapat terjadi pada aplikasi hybrid dengan aplikasi web dan aplikasi Android yang memiliki ID klien Google OAuth 2.0, tetapi memiliki proyek API Google yang sama.

Ketika Anda menulis kebijakan menggunakan kunci kondisi `accounts.google.com:aud`, Anda harus mengetahui apakah aplikasi tersebut merupakan aplikasi hibrida yang menetapkan kolom `azp` tersebut atau bukan.

`azp`Kolom Tidak Ditetapkan

Contoh kebijakan berikut bekerja untuk aplikasi non-hibrida yang tidak menetapkan kolom `azp`. Dalam hal ini, nilai kolom Token ID Google `aud` sesuai dengan nilai kunci kondisi `accounts.google.com:aud` dan `accounts.google.com:oaud`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

Kumpulan Kolom `azp`

Contoh kebijakan berikut berfungsi untuk aplikasi non-hibrida yang tidak menetapkan kolom `azp`. Dalam hal ini, nilai kolom `aud` Token ID Google hanya sesuai dengan nilai kunci kondisi `accounts.google.com:oaud`. Nilai kolom `azp` sesuai dengan nilai kunci kondisi `accounts.google.com:aud`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

surel

Bekerja dengan operator string.

Contoh: `accounts.google.com:email`

Kunci kondisi ini memvalidasi alamat email pengguna. Nilai klaim ini mungkin tidak unik untuk akun ini dan dapat berubah seiring waktu, oleh karena itu Anda tidak boleh menggunakan nilai ini sebagai pengidentifikasi utama untuk memverifikasi catatan pengguna Anda.

oauda

Bekerja dengan operator string.

Contoh: `accounts.google.com:oaud`

Kunci ini menentukan audiens lain (aud) yang dimaksudkan untuk token ID ini. Kunci ini harus salah satu dari ID klien OAuth 2.0 aplikasi Anda.

sub

Bekerja dengan operator string.

Contoh:

- `accounts.google.com:sub`
- token.actions.githubusercontent.com:sub

Gunakan tombol ini untuk memverifikasi bahwa subjek cocok dengan subjek yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

Dalam kebijakan kepercayaan peran berikut, kunci `sub` kondisi membatasi peran ke GitHub cabang bernama`demo`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub": "repo:org-name/repo-
name:ref:refs/heads/demo"
        }
      }
    }
  ]
}
```

Amazon Cognito

Tab ini menjelaskan cara Amazon Cognito memetakan klaim OIDC untuk meng AWS STS kondisikan kunci konteks di. AWS Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

Untuk peran yang digunakan oleh Amazon Cognito, kunci didefinisikan menggunakan `cognito-identity.amazonaws.com` diikuti oleh klaim.

Untuk informasi selengkapnya tentang pemetaan klaim kumpulan identitas, lihat Pemetaan penyedia default di Panduan Pengembang Amazon Cognito. Untuk informasi selengkapnya tentang pemetaan klaim kumpulan pengguna, lihat Menggunakan token ID di Panduan Pengembang Amazon Cognito.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
amr	amr	Ya
aud	aud	Ya
oaud	aud	Tidak
sub	sub	Ya

amr

Bekerja dengan operator string. Kunci ini memiliki banyak nilai, artinya jika Anda mengujinya dalam suatu kebijakan yang menggunakan operator kumpulan kondisi.

Contoh — `cognito-identity.amazonaws.com:amr`

Referensi Metode Otentikasi mencakup informasi login tentang pengguna. Kunci dapat berisi nilai-nilai berikut:

- Jika pengguna tidak autentikasi, kunci hanya berisi `unauthenticated`.
- Jika pengguna diautentikasi, kunci berisi nilai `authenticated` dan nama penyedia login yang digunakan dalam panggilan (`cognito-identity.amazonaws.com`).

Sebagai contoh, kondisi berikut dalam kebijakan kepercayaan untuk peran Amazon Cognito menguji apakah pengguna tidak diautentikasi.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

Aud

Bekerja dengan operator string.

Contoh — `cognito-identity.amazonaws.com:aud`

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito menampilkan nilai yang sama dalam klaim token `client_id` akses.

oauda

Bekerja dengan operator string.

Contoh — `cognito-identity.amazonaws.com:oaud`

Klien aplikasi kumpulan pengguna yang mengautentikasi pengguna Anda. Amazon Cognito menampilkan nilai yang sama dalam klaim token `client_id` akses.

sub

Bekerja dengan operator string.

Contoh — `cognito-identity.amazonaws.com:sub`

Pengidentifikasi unik (UUID), atau subjek, untuk pengguna yang diautentikasi. Nama pengguna mungkin tidak unik di kumpulan pengguna Anda. Sub klaim adalah cara terbaik untuk mengidentifikasi pengguna tertentu. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

```
"Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
```

Login with Amazon

Tab ini menjelaskan cara Login dengan Amazon memetakan klaim OIDC untuk meng AWS STS kondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
app_id	ID Aplikasi	Ya
sub	ID Pengguna	Ya
user_id	ID Pengguna	Ya

app_id

Bekerja dengan operator string.

Contoh — `www.amazon.com:app_id`

Kunci ini menentukan konteks audiens yang cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lainnya.

sub

Bekerja dengan operator string.

Contoh — `www.amazon.com:sub`

Kunci ini memverifikasi bahwa ID pengguna cocok dengan ID yang Anda tentukan dalam kebijakan. Anda dapat menggunakan kunci `sub` dengan kunci `aud` untuk penyedia identitas yang sama.

user_id

Bekerja dengan operator string.

Contoh — `www.amazon.com:user_id`

Kunci ini menentukan konteks audiens yang cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lain.Anda dapat menggunakan `user_id` kunci dengan `id` kunci untuk penyedia identitas yang sama.

Facebook

Tab ini menjelaskan bagaimana Facebook memetakan klaim OIDC untuk meng AWS STS kondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
app_id	ID Aplikasi	Ya
id	id	Ya

app_id

Bekerja dengan operator string.

Contoh — `graph.facebook.com:app_id`

Kunci ini memverifikasi bahwa konteks audiens cocok dengan `aud` bidang yang digunakan oleh penyedia identitas lainnya.

id

Bekerja dengan operator string.

Contoh — `graph.facebook.com:id`

Kunci ini memverifikasi bahwa ID aplikasi (atau situs) cocok dengan ID yang Anda tentukan dalam kebijakan.

GitHub

Tab ini menjelaskan bagaimana Ac GitHub tions memetakan klaim OIDC ke AWS STS kondisi kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
aktor	aktor	Tidak
aktor_id	aktor_id	Tidak
lowjob_workflow_ref	lowjob_workflow_ref	Tidak
repositori	repositori	Tidak
repositori_id	repositori_id	Tidak
id pemilik repositori	id pemilik repositori	Tidak
alur kerja	alur kerja	Tidak
referensi	referensi	Tidak
lingkungan	lingkungan	Tidak
Enterprise_id	Enterprise_id	Tidak

aktor

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:actor`

Kunci ini mengidentifikasi akun pribadi yang memulai proses alur kerja. Gunakan ini untuk membatasi akses ke aktor tertentu.

aktor_id

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:actor_id`

Kunci ini memverifikasi ID akun pribadi yang memulai proses alur kerja. ID aktor dihasilkan oleh GitHub dan tidak dapat diubah.

lowjob_workflow_ref

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:job_workflow_ref`

Kunci ini berisi jalur referensi ke alur kerja yang dapat digunakan kembali untuk pekerjaan menggunakan alur kerja yang dapat digunakan kembali. Gunakan ini untuk membatasi akses ke alur kerja tertentu dan untuk memastikan hanya alur kerja yang disetujui yang dapat mengambil peran.

repositori

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:repository`

Kunci ini mengidentifikasi repositori tempat alur kerja berjalan. Gunakan ini untuk membatasi akses ke GitHub repositori tertentu.

repositori_id

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:repository_id`

Kunci ini memverifikasi ID repositori tempat alur kerja berjalan. ID repositori tidak dapat diubah dan tidak berubah bahkan jika repositori diganti namanya.

id pemilik repositori

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:repository_owner_id`

Kunci ini memverifikasi ID organisasi tempat repositori disimpan. ID pemilik repositori adalah pengenal unik yang stabil yang tidak berubah.

alur kerja

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:workflow`

Kunci ini berisi nama alur kerja. Gunakan ini untuk membatasi akses ke alur kerja tertentu dalam repositori Anda.

referensi

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:ref`

Kunci ini mengidentifikasi git ref (cabang atau tag) yang memicu dijalankan alur kerja. Gunakan ini untuk membatasi akses berdasarkan cabang tertentu, seperti hanya mengizinkan `main` atau `production` cabang.

lingkungan

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:environment`

Kunci ini berisi nama lingkungan yang digunakan oleh pekerjaan. Gunakan ini untuk menerapkan kontrol akses berbasis lingkungan, seperti izin terpisah untuk lingkungan pengembangan, pementasan, dan produksi.

> **Note**
>
> Jika klaim lingkungan disertakan dalam kebijakan kepercayaan Anda, lingkungan harus dikonfigurasi dan disediakan dalam al GitHub ur kerja.

Enterprise_id

Bekerja dengan operator string.

Contoh — `token.actions.githubusercontent.com:enterprise_id`

Kunci ini memverifikasi ID perusahaan yang berisi repositori tempat alur kerja berjalan. Gunakan ini untuk memastikan akses terbatas pada repositori dalam organisasi GitHub Perusahaan Anda.

⚠ Gunakan pengidentifikasi yang tidak dapat diubah, bukan nama

Aktif GitHub, repositori, organisasi, dan nama pengguna dapat berubah. Nama yang dibebaskan dengan mengganti nama atau penghapusan dapat diklaim oleh akun yang berbeda. Kebijakan yang hanya bergantung pada klaim berbasis nama yang dapat berubah (seperti `repository` atau`actor`) dapat memberikan akses ke identitas yang tidak diinginkan. Untuk membantu melindungi terhadap hal ini, gunakan pengidentifikasi yang tidak dapat diubah yang GitHub disediakan, seperti`repository_id`,, atau `repository_owner_id`actor_id, dalam kebijakan kepercayaan peran atau kebijakan kontrol sumber daya Anda.

Contoh kebijakan kepercayaan berikut menggunakan klaim kustom dalam token GitHub OIDC untuk membatasi akses ke peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "StringEquals": {
                "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
                "token.actions.githubusercontent.com:repository_owner_id": "123456",
                "token.actions.githubusercontent.com:repository_id": "1296269",
                "token.actions.githubusercontent.com:actor_id": "1",
                "token.actions.githubusercontent.com:ref": "refs/heads/main",
                "token.actions.githubusercontent.com:enterprise_id": "345"
               }
           }
        }
```

```
    ]
}
```

Google

Tab ini menjelaskan bagaimana Google memetakan OIDC mengklaim meng AWS STS kondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
google/organization_number	google:nomor_organisasi	Tidak

google/organization_number

Bekerja dengan operator numerik.

Contoh — `accounts.google.com:google/organization_number`

Kunci ini memverifikasi bahwa token mewakili identitas Google milik organisasi Google Cloud atau Google Workspace tertentu. Gunakan ini untuk membatasi akses ke pengguna dari organisasi tertentu, memastikan bahwa hanya identitas dari organisasi Anda yang dapat mengambil peran tersebut.

Contoh kebijakan kepercayaan berikut menggunakan `google/organization_number` klaim untuk membatasi akses ke peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "accounts.google.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "NumericEquals": {
                "accounts.google.com:google/organization_number": "123456"
```

```
                }
            }
        }
    ]
}
```

CircleCI

Tab ini menjelaskan bagaimana CircleCI memetakan klaim OIDC ke AWS STS kondisi kunci konteks di. AWS Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
idc.circleci. com/project-id	idc.circleci. com/project-id	Tidak

idc.circleci. com/project-id

Bekerja dengan operator string.

Contoh — `circleci-issuer-url:oidc.circleci.com/project-id`

Kunci ini mengidentifikasi proyek CircleCI tempat pekerjaan berjalan. Nilainya adalah string yang berisi UUID yang secara unik mengidentifikasi proyek CircleCI. Gunakan ini untuk membatasi akses ke proyek CircleCI tertentu.

Contoh kebijakan kepercayaan berikut menggunakan `oidc.circleci.com/project-id` klaim untuk membatasi akses ke peran.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/oidc.circleci.com/
org/12345"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringEquals": {
          "oidc.circleci.com/org/12345:aud": "sts.amazonaws.com",
          "oidc.circleci.com/org/12345:oidc.circleci.com/project-id": "76543210-
ba98-fedc-3210-edcba0987654"
        }
      }
    }
  ]
}
```

Oracle Cloud Infrastructure (OCI)

Tab ini menjelaskan cara Oracle Cloud Infrastructure memetakan klaim OIDC ke AWS STS kondisi kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
rpst_id	rpst_id	Tidak

rpst_id

Bekerja dengan operator string.

Contoh — `oci-issuer-url:rpst_id`

Kunci ini secara unik mengidentifikasi prinsip sumber daya di OCI. Gunakan ini untuk membatasi akses ke prinsipal sumber daya OCI tertentu. rpst_id (Resource Principal Session Token ID) menyediakan pengenal stabil untuk otentikasi berbasis sumber daya OCI.

Contoh kebijakan kepercayaan berikut menggunakan `rpst_id` klaim untuk membatasi akses ke peran.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::123456789012:oidc-provider/idcs-
abc123ef5678901234abcd.identity.oraclecloud.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:aud":
 "sts.amazonaws.com",
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:rpst_id": "your-
rpst-id"
        }
      }
    }
  ]
}
```

GitLab.com

Tab ini menjelaskan bagaimana memet GitLab.com CI/CD akan klaim OIDC untuk meng AWS STS kondisikan kunci konteks di AWS. Anda dapat menggunakan kunci ini untuk mengontrol akses ke sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

> **Note**
>
> Kunci kondisi khusus ini hanya didukung saat menggunakan instance SaaS GitLab.com multi-tenant sebagai penyedia identitas OIDC. Anda tidak dapat menggunakan kunci kondisi ini dengan D GitLab edicated atau GitLab Self-Managed instance.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
namespace_id	namespace_id	Tidak
Proyek_id	Proyek_id	Tidak
user_id	user_id	Tidak
login pengguna	login pengguna	Tidak
email_pengguna	email_pengguna	Tidak

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
tingkat_akses_pengguna	tingkat_akses_pengguna	Tidak
ref_dilindungi	ref_dilindungi	Tidak
Pipeline_source	Pipeline_source	Tidak
lingkungan pelari	lingkungan pelari	Tidak

namespace_id

Bekerja dengan operator string.

Contoh — `gitlab.com:namespace_id`

Kunci ini mengidentifikasi ID namespace (grup) dari proyek yang menjalankan CI/CD pekerjaan. ID namespace adalah pengidentifikasi unik yang stabil yang tidak berubah jika grup diganti namanya.

Proyek_id

Bekerja dengan operator string.

Contoh — `gitlab.com:project_id`

Kunci ini mengidentifikasi ID proyek yang menjalankan CI/CD pekerjaan. ID proyek adalah pengidentifikasi unik yang stabil yang tidak berubah jika proyek diganti namanya atau dipindahkan.

user_id

Bekerja dengan operator string.

Contoh — `gitlab.com:user_id`

Kunci ini mengidentifikasi ID pengguna yang menjalankan CI/CD pekerjaan. ID pengguna adalah pengidentifikasi unik yang stabil.

login pengguna

Bekerja dengan operator string.

Contoh — `gitlab.com:user_login`

Kunci ini mengidentifikasi nama pengguna yang menjalankan CI/CD pekerjaan. Nama pengguna dapat diubah; untuk pengenal yang stabil, gunakan sebagai `user_id` gantinya.

email_pengguna

Bekerja dengan operator string.

Contoh — `gitlab.com:user_email`

Kunci ini mengidentifikasi alamat email pengguna yang menjalankan CI/CD pekerjaan. Nilai klaim ini mungkin berubah seiring waktu; untuk pengenal stabil, gunakan sebagai `user_id` gantinya.

tingkat_akses_pengguna

Bekerja dengan operator string.

Contoh — `gitlab.com:user_access_level`

Kunci ini mengidentifikasi tingkat akses pengguna dalam proyek (misalnya,, `maintainer`,`developer`,`owner`).

ref_dilindungi

Bekerja dengan operator string.

Contoh — `gitlab.com:ref_protected`

Kunci ini menunjukkan apakah referensi Git yang memicu pekerjaan dilindungi. Nilainya `true` jika dilindungi, `false` sebaliknya.

Pipeline_source

Bekerja dengan operator string.

Contoh — `gitlab.com:pipeline_source`

Kunci ini mengidentifikasi sumber yang memicu pipeline (misalnya,, `pushweb`,`schedule`,`api`,`merge_request_event`).

runner_environment

Bekerja dengan operator string.

Contoh — `gitlab.com:runner_environment`

Kunci ini mengidentifikasi jenis pelari yang digunakan oleh CI/CD pekerjaan.

> **Important**
>
> Di GitLab.com SaaS, jalur grup atau proyek yang dihapus sebelumnya dapat dibuat ulang oleh pengguna yang berbeda, yang dapat menyebabkan kebijakan hanya mengandalkan klaim berbasis jalur (seperti`sub`) untuk memberikan akses ke identitas yang tidak diinginkan. Pada 1 Juni 2026, GitLab telah menerapkan pembaruan sisi platform yang mencegah token CI ID baru dikeluarkan untuk jalur proyek yang sebelumnya mengeluarkan token, mengurangi risiko ini ke depan. Sebagai lapisan perlindungan tambahan, AWS dan GitLab merekomendasikan untuk menyertakan kondisi pada pengidentifikasi unik yang stabil — seperti `namespace_id` atau `project_id` — dalam kebijakan kepercayaan peran atau kebijakan kontrol sumber daya Anda.

Contoh kebijakan kepercayaan berikut menggunakan `namespace_id` klaim bersama `sub` untuk membatasi akses ke GitLab.com proyek dan cabang tertentu.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/gitlab.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "gitlab.com:sub": "project_path:mygroup/
myproject:ref_type:branch:ref:main",
          "gitlab.com:namespace_id": "12345"
        }
      }
    }
  ]
}
```

Buildkite

Tab ini menjelaskan bagaimana Buildkite memetakan klaim OIDC untuk meng AWS STS kondisikan kunci konteks di. AWS Anda dapat menggunakan kunci ini untuk mengontrol akses ke

sebuah peran. Untuk melakukan itu, bandingkan kunci AWS STS kondisi dengan nilai di kolom klaim JWT IdP.

AWS STS kunci kondisi	Klaim IdP JWT	Tersedia dalam sesi
organisation_slug	organisation_slug	Tidak
organisation_id	organisation_id	Tidak
pipeline_slug	pipeline_slug	Tidak
_id pipa	_id pipa	Tidak
nama klaster	nama klaster	Tidak
Cluster_id	Cluster_id	Tidak
bangun_cabang	bangun_cabang	Tidak

organisation_slug

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:organization_slug`

Kunci ini mengidentifikasi organisasi Buildkite tempat pekerjaan berjalan. Gunakan ini untuk membatasi akses ke organisasi tertentu.

organisation_id

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:organization_id`

Kunci ini memverifikasi ID organisasi Buildkite tempat pekerjaan berjalan. ID organisasi dihasilkan oleh Buildkite dan tidak dapat diubah.

Pipeline_slug

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:pipeline_slug`

Kunci ini mengidentifikasi pipeline tempat pekerjaan berjalan. Gunakan ini untuk membatasi akses ke pipeline Buildkite tertentu.

_id pipa

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:pipeline_id`

Kunci ini memverifikasi ID pipeline tempat pekerjaan berjalan. ID pipa dihasilkan oleh Buildkite dan tidak dapat diubah.

nama klaster

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:cluster_name`

Kunci ini mengidentifikasi cluster tempat pekerjaan berjalan. Gunakan ini untuk membatasi akses ke cluster Buildkite tertentu.

Cluster_id

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:cluster_id`

Kunci ini memverifikasi ID cluster tempat pekerjaan berjalan. ID cluster dihasilkan oleh Buildkite dan tidak dapat diubah.

bangun_cabang

Bekerja dengan operator string.

Contoh — `agent.buildkite.com:build_branch`

Kunci ini mengidentifikasi cabang git yang memicu build. Gunakan ini untuk membatasi akses berdasarkan cabang tertentu, seperti hanya mengizinkan `main` atau `production` cabang.

⚠ Gunakan pengidentifikasi yang tidak dapat diubah, bukan siput

Di Buildkite, organisasi yang dihapus atau slug pipeline dapat segera didaftarkan ulang oleh organisasi atau pipeline yang berbeda. Buildkite mengeluarkan token OIDC untuk

siput ini dari penerbit yang sama (). `agent.buildkite.com` Akibatnya, kebijakan yang hanya mengandalkan klaim berbasis siput (seperti `organization_slug` ataupipeline_slug) dapat memberikan akses ke identitas yang tidak diinginkan. Untuk membantu melindungi dari hal ini, gunakan pengidentifikasi yang stabil dan tidak dapat diubah, seperti `organization_id` ataupipeline_id, dalam kebijakan kepercayaan peran atau kebijakan kontrol sumber daya Anda.

Contoh kebijakan kepercayaan berikut menggunakan klaim khusus di token Buildkite OIDC untuk membatasi akses ke peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::123456789012:oidc-provider/
agent.buildkite.com"
        },
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {
                "agent.buildkite.com:aud": "sts.amazonaws.com",
                "agent.buildkite.com:organization_id": "0191e7a0-1234-7abc-
def0-123456789abc",
                "agent.buildkite.com:pipeline_id": "0191e7a1-5678-7abc-
def0-123456789abc",
                "agent.buildkite.com:build_branch": "main"
            }
        }
    }
    ]
}
```

Informasi lebih lanjut tentang Federasi OIDC

- Panduan Pengguna Amazon Cognito
- Federasi OIDC

Kunci yang tersedia untuk SAML-based AWS STS federasi

Jika Anda bekerja dengan SAML-based federasi menggunakan AWS Security Token Service (AWS STS), Anda dapat menyertakan kunci kondisi tambahan dalam kebijakan.

Kebijakan kepercayaan peran SAML

Dalam kebijakan kepercayaan dari suatu peran, Anda dapat menyertakan kunci-kunci berikut yang membantu Anda menentukan apakah pemanggil diperbolehkan untuk memegang peran tersebut. Kecuali untuk `saml:doc`, semua nilai berasal dari pernyataan SAML. Semua item dalam daftar tersedia di editor visual konsol IAM saat Anda membuat atau mengedit kebijakan dengan kondisi. Item ditandai denganx `[]` dapat memiliki nilai yang merupakan daftar jenis yang ditentukan.

sml:aud

Bekerja dengan operator string.

URL titik akhir tempat pernyataan SAML disajikan. Nilai untuk kunci ini berasal dari kolom `SAML Recipient` dalam pernyataan, bukan kolom `Audience`.

saml:commonName[]

Bekerja dengan operator string.

Ini adalah atribut `commonName`.

saml:cn[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:doc

Bekerja dengan operator string.

Ini mewakili prinsipal yang digunakan untuk menerima peran tersebut. Formatnya adalah*`account-ID`*/*`provider-friendly-name`*, seperti`123456789012/SAMLProviderName`. Nilai account-ID merujuk pada akun yang memiliki penyedia SAML.

saml:edupersonaffiliation[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonassurance[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonentitlement[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonnickname[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonorgdn

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonorgunitdn[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprimaryaffiliation

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprimaryorgunitdn

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonprincipalname

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersonscopedaffiliation[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:edupersontargetedid[]

Bekerja dengan operator string.

Ini adalah atribut `eduPerson`.

saml:eduorghomepageuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorglegalname[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgsuperioruri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:eduorgwhitepagesuri[]

Bekerja dengan operator string.

Ini adalah atribut `eduOrg`.

saml:givenName[]

Bekerja dengan operator string.

Ini adalah atribut `givenName`.

sml:iss

Bekerja dengan operator string.

Penerbit yang diwakili oleh URN.

saml:mail[]

Bekerja dengan operator string.

Ini adalah atribut `mail`.

saml:name[]

Bekerja dengan operator string.

Ini adalah atribut `name`.

saml:namequalifier

Bekerja dengan operator string.

Nilai hash berdasarkan nama yang penyedia SAML yang dikenal. Nilai adalah rangkaian dari nilai-nilai berikut, diurutkan dan dipisahkan oleh karakter '/':

1. Nilai tanggapan `Issuer` (`saml:iss`)
2. ID akun `AWS`
3. Nama yang dikenal (bagian terakhir ARN) penyedia SAML di IAM

Gabungan ID akun dan nama ramah penyedia SAML tersedia untuk kebijakan IAM sebagai kuncinya. `saml:doc` Untuk informasi selengkapnya, lihat Mengidentifikasi pengguna secara unik di SAML-based federasi.

saml:organizationStatus[]

Bekerja dengan operator string.

Ini adalah atribut `organizationStatus`.

saml:primaryGroupSID[]

Bekerja dengan operator string.

Ini adalah atribut `primaryGroupSID`.

saml:sub

Bekerja dengan operator string.

Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna individu dalam suatu organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Bekerja dengan operator string.

Kunci ini dapat memiliki nilai `persistent`, `transient`, atau terdiri atas URL `Format` penuh dari elemen `Subject` dan `NameID` yang digunakan dalam pernyataan SAML Anda. Nilai dari `persistent` menunjukkan bahwa nilai dalam `saml:sub` sama untuk pengguna di antara sesi. Jika nilainya `transient`, pengguna memiliki nilai `saml:sub` untuk setiap sesi. Untuk informasi tentang elemen `NameID` `Format` atribut, lihat Mengonfigurasikan pernyataan SAML untuk respons autentikasi.

saml:surname[]

Bekerja dengan operator string.

Ini adalah atribut `surnameuid`.

saml:uid[]

Bekerja dengan operator string.

Ini adalah atribut `uid`.

saml:x500 [] UniqueIdentifier

Bekerja dengan operator string.

Ini adalah atribut `x500UniqueIdentifier`.

Untuk informasi umum tentang atribut `eduPerson` dan `eduOrg`, lihat situs web Wiki REFEDS. Untuk daftar atribut `eduPerson`, lihat Spesifikasi Kelas Objek eduPerson (201602).

Kunci kondisi dengan tipe berupa daftar dapat mencakup beberapa nilai. Untuk membuat kondisi dalam kebijakan untuk nilai-nilai daftar, Anda dapat menggunakan operator kumpulan (`ForAllValues`, `ForAnyValue`). Misalnya, untuk mengizinkan pengguna yang berafiliasi dengan "fakultas" atau "staf" (tetapi bukan "mahasiswa"), Anda dapat menggunakan ketentuan seperti berikut:

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Cross-service SAML-based AWS STS kunci konteks federasi

Beberapa kunci kondisi SAML-based federasi dapat digunakan dalam permintaan berikutnya untuk mengotorisasi AWS operasi di layanan dan `AssumeRole` panggilan lain. Ini adalah kunci kondisi berikut yang dapat digunakan dalam kebijakan kepercayaan peran ketika prinsipal federasi mengambil peran lain, dan dalam kebijakan sumber daya dari AWS layanan lain untuk mengotorisasi akses sumber daya oleh prinsipal federasi. Untuk informasi selengkapnya tentang menggunakan kunci ini, lihat Tentang federasi berbasis SAML 2.0.

Pilih kunci kondisi untuk melihat deskripsi.

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> Tidak ada kunci kondisi SAML-based federasi lain yang tersedia untuk digunakan setelah respons otentikasi penyedia identitas eksternal (IdP) awal.

Kunci yang tersedia untuk AWS STS

Anda dapat menggunakan kunci kondisi berikut dalam kebijakan kepercayaan peran IAM untuk peran yang diasumsikan menggunakan operasi AWS Security Token Service (AWS STS).

saml:sub

Bekerja dengan operator string.

Ini adalah subjek klaim, yang mencakup nilai yang secara unik mengidentifikasi pengguna individu dalam suatu organisasi (misalnya, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sts: AWSServiceName

Bekerja dengan operator string.

Gunakan kunci ini untuk menentukan layanan di mana token pembawa dapat digunakan. Saat Anda menggunakan kunci kondisi ini dalam suatu kebijakan, tentukan layanan menggunakan prinsipal layanan. Prinsipal layanan adalah nama layanan yang dapat ditentukan dalam elemen

`Principal` dari suatu kebijakan. Misalnya, `codeartifact.amazonaws.com` adalah prinsi AWS CodeArtifact pal layanan.

Ketersediaan – Kunci ini tersedia dalam permintaan yang mendapatkan token pembawa. Anda tidak dapat melakukan panggilan langsung AWS STS untuk mendapatkan token pembawa. Saat Anda melakukan beberapa operasi dalam layanan lain, layanan meminta token pembawa atas nama Anda.

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Misalnya, AWS CodeArtifact mengharuskan prinsipal untuk menggunakan token pembawa untuk melakukan beberapa operasi. Perintah `aws codeartifact get-authorization-token` mengembalikan token pembawa. Anda kemudian dapat menggunakan token pembawa untuk melakukan AWS CodeArtifact operasi. Untuk informasi selengkapnya tentang token pembawa, lihat Token pembawa layanan.

Anda dapat menggunakan kunci kondisi ini untuk mengizinkan prinsipal mendapatkan token pembawa untuk digunakan dengan layanan tertentu.

sts: DurationSeconds

Bekerja dengan operator numerik.

Gunakan kunci ini untuk menentukan durasi (dalam detik) yang dapat digunakan kepala sekolah saat mendapatkan token AWS AWS STS pembawa atau Token Web JSON dari GetWebIdentityToken API.

Ketersediaan — Kunci ini hadir dalam permintaan yang mendapatkan token pembawa atau Token Web JSON dari GetWebIdentityToken API. Anda tidak dapat melakukan panggilan langsung AWS STS untuk mendapatkan token pembawa. Saat Anda melakukan beberapa operasi dalam layanan lain, layanan meminta token pembawa atas nama Anda. Kuncinya tidak berlaku untuk operasi AWS STS asumsi-peran.

Beberapa AWS layanan mengharuskan Anda memiliki izin untuk mendapatkan token pembawa AWS STS layanan sebelum Anda dapat mengakses sumber daya mereka secara terprogram. Misalnya, AWS CodeArtifact mengharuskan prinsipal untuk menggunakan token pembawa untuk melakukan beberapa operasi. Perintah `aws codeartifact get-authorization-token` mengembalikan token pembawa. Anda kemudian dapat menggunakan token pembawa untuk melakukan AWS CodeArtifact operasi. Untuk informasi selengkapnya tentang token pembawa, lihat Token pembawa layanan.

sts: IdentityTokenAudience

Bekerja dengan operator string.

Gunakan kunci ini untuk menentukan audiens di mana kepala sekolah IAM dapat meminta Token Web JSON (JWT) menggunakan API. GetWebIdentityToken Jika kunci kondisi ini ada dalam kebijakan IAM, kepala sekolah IAM hanya dapat meminta token untuk audiens yang ditentukan dalam kebijakan. Layanan eksternal memvalidasi klaim audiens (“aud”) di Token Web JSON untuk memastikan token ditujukan untuk mereka.

Ketersediaan — Kunci ini hadir dalam permintaan ke GetWebIdentityToken API yang digunakan untuk mendapatkan Token Web JSON (JWT) untuk otentikasi dengan layanan eksternal.

Saat Anda menggunakan kunci kondisi ini dalam kebijakan, tentukan nilai audiens yang cocok dengan pengenal penerima yang dituju (misalnya,https://api.example.com).

Kebijakan contoh berikut memungkinkan prinsipal meminta token untuk layanan eksternal yang ditentukan:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api2.example.com",
                        "https://api1.example.com"
                    ]
                }
            }
        }
    ]
}
```

sts: SigningAlgorithm

Bekerja dengan operator string.

Gunakan kunci ini untuk menentukan algoritma kriptografi yang digunakan AWS AWS STS untuk menandatangani Token Web JSON (JWT) yang dihasilkan oleh API. GetWebIdentityToken Saat Anda menggunakan kunci kondisi ini dalam kebijakan, tentukan ES384 (ECDSA dengan P-384 kurva dan SHA-384) atau RS256 (RSA dengan). SHA-256

Ketersediaan — Kunci ini hadir dalam permintaan ke GetWebIdentityToken API yang digunakan untuk mendapatkan Token Web JSON (JWT) untuk otentikasi dengan layanan eksternal.

Anda dapat menggunakan kunci kondisi ini untuk memaksakan bahwa prinsipal IAM meminta token menggunakan algoritma penandatanganan yang kompatibel dengan persyaratan keamanan Anda atau layanan eksternal yang Anda integrasikan. ES384 memberikan keamanan dan kinerja yang optimal, sementara RS256 menawarkan kompatibilitas yang lebih luas dengan sistem yang tidak mendukung ECDSA.

Kebijakan contoh berikut mengharuskan prinsipal menggunakan algoritma penandatanganan ES384:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

sts: ExternalId

Bekerja dengan operator string.

Gunakan kunci ini untuk mengharuskan kepala sekolah memberikan pengenal khusus saat mengambil peran IAM.

Ketersediaan — Kunci ini hadir dalam permintaan ketika kepala sekolah menyediakan ID eksternal saat mengambil peran menggunakan AWS API AWS CLI or.

Pengidentifikasi unik yang mungkin diperlukan saat Anda menerima peran dalam akun lain. Jika administrator akun yang memiliki peran tersebut memberi Anda ID eksternal, berikan nilai tersebut di parameter `ExternalId`. Nilai ini dapat berupa string, seperti frasa sandi atau nomor akun. Fungsi utama dari ID eksternal adalah mengatasi dan mencegah masalah deputi yang membingungkan. Untuk informasi selengkapnya tentang ID eksternal dan masalah deputi yang membingungkan, lihat Akses ke Akun AWS dimiliki oleh pihak ketiga.

Nilai `ExternalId` harus memiliki minimal 2 karakter dan maksimal 1.224 karakter. Nilai harus berupa alfanumerik tanpa spasi. Alfanumerik dapat mencakup simbol berikut: plus (+), sama (=), koma (,), titik (.), di (@), titik dua (:), garis miring (/), dan tanda hubung (-).

sts:RequestContext/kunci konteks

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan pasangan kunci-nilai konteks sesi yang disematkan dalam pernyataan konteks yang ditandatangani penerbit token tepercaya yang diteruskan dalam permintaan dengan nilai kunci konteks yang ditentukan dalam kebijakan kepercayaan peran.

Ketersediaan — Kunci ini hadir dalam permintaan ketika pernyataan konteks disediakan dalam parameter `ProvidedContexts` permintaan saat mengambil peran menggunakan operasi AWS STS AssumeRole API.

Kunci konteks ini diformat sebagai `"sts:RequestContext/context-key":"context-value"` where `context-key` dan `context-value` merupakan pasangan kunci-nilai konteks. Ketika beberapa kunci konteks disematkan dalam pernyataan konteks yang ditandatangani yang diteruskan dalam permintaan, ada satu kunci konteks untuk setiap pasangan kunci-nilai. Anda harus memberikan izin untuk `sts:SetContext` tindakan dalam kebijakan kepercayaan peran untuk mengizinkan prinsipal menetapkan kunci konteks dalam token sesi yang dihasilkan. Untuk mempelajari lebih lanjut tentang kunci konteks Pusat Identitas IAM yang didukung yang dapat digunakan dengan kunci ini, lihat kunci AWS STS kondisi untuk Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk menerapkan kontrol akses berbutir halus berdasarkan pengguna atau atributnya saat mereka mengambil peran. Setelah peran diasumsikan, aktivitas muncul di AWS CloudTrail log di dalam `AdditionalEventData` atribut, berisi pasangan kunci-nilai konteks sesi yang ditetapkan oleh penyedia konteks dalam permintaan peran asumsi. Ini memudahkan administrator membedakan di antara sesi peran saat peran digunakan oleh prinsipal yang berbeda. Pasangan kunci-nilai diatur oleh penyedia konteks yang ditentukan, bukan oleh AWS CloudTrail atau. AWS STS Ini

memberikan kontrol penyedia konteks atas konteks apa yang disertakan dalam CloudTrail log dan informasi sesi.

sts: RequestContextProviders

Bekerja IT Operator ARN .

Gunakan kunci ini untuk membandingkan penyedia konteks ARN dalam permintaan dengan penyedia konteks ARN yang ditentukan dalam kebijakan kepercayaan peran.

Ketersediaan — Kunci ini hadir dalam permintaan ketika pernyataan konteks disediakan dalam parameter `ProvidedContexts` permintaan saat mengambil peran menggunakan operasi AWS STS AssumeRole API.

Contoh kondisi berikut memeriksa apakah penyedia konteks ARN yang diteruskan dalam permintaan cocok dengan ARN yang ditentukan dalam kondisi kebijakan kepercayaan peran. Kami sarankan Anda menambahkan pemeriksaan nol dengan `ForAllValues` untuk mencegah kunci konteks yang hilang atau kunci konteks dengan nilai kosong dievaluasi menjadi true. Lihat perinciannya di Operator ketentuan memeriksa keberadaan kunci kondisi.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Action": "sts:SetContext",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
      "ForAllValues:ArnEquals": {
        "sts:RequestContextProviders": [
          "arn:aws:iam::aws:contextProvider/IdentityCenter"
        ]
      },
      "Null": {
        "sts:RequestContextProviders": "false"
      }
    }
  }
}
```

sts: RoleAuthorizedByIdp

Bekerja dengan operatorBoolean.

Gunakan kunci ini untuk memverifikasi bahwa penyedia identitas (IdP) secara eksplisit mengizinkan peran yang diminta melalui `https://aws.amazon.com/roles` klaim di token OIDC. Klaim peran berisi ARN peran IAM yang diizinkan untuk diasumsikan oleh token. AWS STS mendukung dua format untuk nilai klaim:

Array JSON (disarankan):

```
"https://aws.amazon.com/roles": [
  "arn:aws:iam::111122223333:role/RoleA",
  "arn:aws:iam::111122223333:role/RoleB"
]
```

Semicolon-delimited string:

```
"https://aws.amazon.com/roles": "arn:aws:iam::111122223333:role/
RoleA;arn:aws:iam::111122223333:role/RoleB"
```

Ketersediaan — Kunci ini hadir dalam permintaan yang menggunakan `AssumeRoleWithWebIdentity` operasi. Nilainya adalah `true` ketika token menyertakan `https://aws.amazon.com/roles` klaim dan peran ARN yang diminta cocok dengan nilai dalam klaim. Nilainya adalah `false` ketika token tidak menyertakan klaim. Karena kunci selalu ada dalam `AssumeRoleWithWebIdentity` permintaan, gunakan operator `Bool` kondisi daripada`BoolIfExists`.

Ketika token menyertakan klaim tetapi peran ARN yang diminta tidak cocok dengan nilai dalam klaim, `AssumeRoleWithWebIdentity` panggilan mengembalikan `InvalidIdentityToken` kesalahan.

Anda dapat menggunakan kunci kondisi ini dalam kebijakan kepercayaan peran, kebijakan kontrol sumber daya (RCP), dan kebijakan titik akhir VPC untuk mewajibkan bahwa IdP mengotorisasi peran sebelum dapat diasumsikan.

Kebijakan kepercayaan peran berikut `AssumeRoleWithWebIdentity` hanya mengizinkan jika IdP mengizinkan peran yang diminta dalam token.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": { "idp.example.com:aud": "my-app-id" },
                "Bool": { "sts:RoleAuthorizedByIdp": "true" }
            }
        }
    ]
}
```

sts: RoleSessionName

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan nama sesi yang ditentukan oleh prinsipal saat mengambil peran dengan nilai yang ditentukan dalam kebijakan.

Ketersediaan — Kunci ini hadir dalam permintaan ketika kepala sekolah mengambil peran menggunakan Konsol Manajemen AWS, perintah CLI asumme-role apa pun, atau operasi API apa pun AWS STS `AssumeRole`.

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk mewajibkan pengguna Anda memberikan nama sesi tertentu saat mereka mengambil peran. Misalnya, Anda dapat meminta pengguna IAM untuk menentukan nama pengguna mereka sendiri sebagai nama sesi mereka. Setelah pengguna IAM mengambil peran, aktivitas muncul di AWS CloudTrail log dengan nama sesi yang cocok dengan nama pengguna mereka. Ini memudahkan administrator membedakan di antara sesi peran saat peran digunakan oleh prinsipal yang berbeda.

Kebijakan kepercayaan peran berikut mengharuskan pengguna IAM di akun `111122223333` memberikan nama pengguna IAM mereka sebagai nama sesi saat mereka mengambil peran tersebut. Persyaratan ini diberlakukan menggunakan `aws:username` variabel kondisi di dalam kunci kondisi. Pernyataan memungkinkan pengguna IAM untuk mengasumsikan peran yang diberlakukan pada kebijakan tersebut. Kebijakan ini tidak mengizinkan siapa pun yang menggunakan kredensial sementara untuk mengambil peran tersebut karena variabel `username` hanya tersedia untuk pengguna IAM.

> **Important**
>
> Anda dapat menggunakan kunci kondisi bernilai tunggal sebagai variabel. Anda tidak dapat menggunakan kunci kondisi multivalue sebagai variabel.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "prefix-
${aws:username}"}
            }
        }
    ]
}
```

Ketika administrator melihat AWS CloudTrail log untuk tindakan, mereka dapat membandingkan nama sesi dengan nama pengguna di akun mereka. Dalam contoh berikut, pengguna bernama `matjac` melakukan operasi menggunakan peran bernama `MateoRole`. Administrator selanjutnya dapat menghubungi Mateo Jackson, yang memiliki pengguna bernama `matjac`.

```
"assumedRoleUser": {
    "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
    "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
}
```

Jika Anda mengizinkan akses lintas akun menggunakan peran, maka pengguna di satu akun dapat mengambil peran di akun lain. ARN dari pengguna peran yang diasumsikan yang tercantum dalam CloudTrail termasuk akun jika peran tersebut ada. Ini tidak termasuk akun pengguna yang mengambil peran tersebut. Pengguna hanya bersifat unik di dalam suatu akun. Oleh karena itu, sebaiknya gunakan metode ini untuk memeriksa CloudTrail log hanya untuk peran yang

diasumsikan oleh pengguna di akun yang Anda kelola. Pengguna Anda mungkin menggunakan nama pengguna yang sama dalam beberapa akun.

sts: SourceIdentity

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan identitas sumber yang ditentukan oleh prinsipal saat mengambil peran dengan nilai yang ditentukan dalam kebijakan.

Ketersediaan — Kunci ini hadir dalam permintaan ketika kepala sekolah menyediakan identitas sumber saat mengambil peran menggunakan perintah asumsi- AWS STS role CLI, atau AWS STS `AssumeRole` operasi API.

Anda dapat menggunakan kunci ini dalam kebijakan kepercayaan peran untuk mewajibkan pengguna Anda menetapkan identitas sumber tertentu saat mereka mengambil peran. Misalnya, Anda dapat meminta tenaga kerja Anda atau identitas gabungan untuk menentukan nilai identitas sumber. Anda dapat mengonfigurasi penyedia identitas Anda (IdP) untuk menggunakan salah satu atribut yang terkait dengan pengguna Anda, seperti nama pengguna atau email sebagai identitas sumber. IdP kemudian meneruskan identitas sumber sebagai atribut dalam pernyataan atau klaim yang dikirimkannya. AWS Nilai atribut identitas sumber mengidentifikasi pengguna atau aplikasi yang mengambil peran.

Setelah pengguna mengambil peran, aktivitas muncul di log AWS CloudTrail dengan nilai identitas sumber yang ditetapkan. Hal ini memudahkan administrator untuk menentukan siapa atau apa yang melakukan tindakan dengan peran AWS. Anda harus memberikan izin untuk tindakan `sts:SetSourceIdentity` untuk mengizinkan identitas menetapkan identitas sumber.

Tidak seperti `sts:RoleSessionName`, setelah identitas sumber diatur, nilai tidak dapat diubah. Ini terdapat dalam konteks permintaan untuk semua tindakan yang diambil dengan peran menggunakan identitas sumber daya. Nilai tetap ada dalam sesi peran berikutnya saat Anda menggunakan kredensial sesi untuk mengambil peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Anda dapat menggunakan kunci kondisi `aws:SourceIdentity` global untuk mengontrol lebih lanjut akses ke AWS sumber daya berdasarkan nilai identitas sumber dalam permintaan berikutnya.

Kebijakan kepercayaan peran berikut memungkinkan pengguna IAM `AdminUser` untuk mengambil peran dalam akun`111122223333`. Ini juga memberikan izin ke `AdminUser` untuk mengatur identitas sumber, selama identitas sumber yang ditetapkan adalah `DiegoRamirez`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Untuk informasi selengkapnya tentang menggunakan informasi identitas sumber, lihat Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan.

sts: TaskPolicyArn

Bekerja IT Operator ARN .

Gunakan kunci ini untuk membandingkan kebijakan ARN dalam AssumeRoot permintaan sts: dengan kebijakan ARN yang ditentukan dalam kebijakan.

Ketersediaan — Kunci ini hadir dalam permintaan saat Anda membuat permintaan menggunakan sts: AssumeRoot.

Administrator dapat menggunakan kunci kondisi ini dalam kebijakan IAM untuk membatasi peran atau pengguna tertentu dalam akun manajemen atau akun administrator yang didelegasikan agar tidak melakukan tindakan tertentu saat mengasumsikan kredentif root. Untuk informasi selengkapnya, lihat Melakukan tugas istimewa pada AWS Organizations akun anggota.

sts: TransitiveTagKeys

Bekerja dengan operator string.

Gunakan kunci ini untuk membandingkan kunci tag sesi transitif dalam permintaan dengan yang ditentukan dalam kebijakan.

Ketersediaan – Kunci ini tersedia dalam permintaan saat Anda mengajukan permintaan menggunakan kredensial keamanan sementara. Ini termasuk kredensial yang dibuat menggunakan operasi asumsi peran, atau operasi `GetFederationToken`.

Saat Anda menggunakan kredensial sesi untuk membuat permintaan berikutnya, konteks permintaan mencakup kunci konteks `aws:PrincipalTag`. Kunci ini mencakup daftar tanda sesi, tanda sesi transitif, dan tanda peran. Tanda sesi transitif adalah tanda yang tetap ada di semua sesi berikutnya saat Anda menggunakan kredensial sesi untuk menjalankan peran lain. Mengasumsikan satu peran dari peran lain disebut rantai peran.

Anda dapat menggunakan kunci kondisi ini dalam kebijakan untuk mewajibkan pengaturan tanda sesi tertentu sebagai transitif saat mengambil peran atau menggabungkan pengguna.

Tindakan, sumber daya, dan kunci kondisi untuk AWS layanan

Setiap AWS layanan dapat menentukan tindakan, sumber daya, dan kunci konteks kondisi untuk digunakan dalam kebijakan IAM. Untuk daftar AWS layanan dan tindakan, sumber daya, dan kunci konteks kondisinya, lihat Tindakan, sumber daya, dan kunci kondisi di Referensi Otorisasi Layanan.

Sumber daya tentang IAM yang dapat dipelajari lebih lanjut

IAM adalah produk yang kaya, dan Anda akan menemukan banyak sumber daya untuk membantu Anda mempelajari lebih lanjut tentang bagaimana IAM dapat membantu Anda mengamankan sumber daya Anda Akun AWS .

Topik

- Identitas
- Kredensial (RDS, access key, dan perangkat MFA)
- Izin dan kebijakan
- Federasi dan delegasi
- IAM dan lainnya AWS produk
- Praktik kerahasiaan umum
- Sumber daya umum

Identitas

Konsultasikan sumber daya ini untuk membuat, mengelola, dan menggunakan identitas.

- Mengelola identitas di Pusat Identitas IAM — Informasi prosedural tentang membuat pengguna dan grup di Pusat Identitas IAM.
- Identitas IAM— Diskusi mendalam tentang pengguna, kelompok, dan peran.

Kredensial (RDS, access key, dan perangkat MFA)

Tinjau panduan berikut untuk mengelola kata sandi, kunci akses, dan perangkat MFA untuk Anda Akun AWS dan pengguna IAM.

- Kata sandi pengguna di AWS— Menjelaskan opsi untuk mengelola kata sandi untuk pengguna IAM di akun Anda.
- Kelola access key untuk pengguna IAM— Menjelaskan bagaimana kunci akses bekerja dan bagaimana Anda dapat menggunakannya untuk melakukan panggilan terprogram ke AWS. Ada alternatif lain yang lebih aman untuk mengakses kunci yang kami sarankan Anda pertimbangkan

terlebih dahulu. Untuk informasi selengkapnya, lihat Pertimbangan dan alternatif untuk kunci akses jangka panjang dalam Referensi Umum AWS panduan ini.

- AWS Multi-factor otentikasi di IAM – Menjelaskan cara mengonfigurasi akun Anda dan pengguna IAM untuk meminta kata sandi dan kode penggunaan sekali pakai yang dibuat di perangkat sebelum diizinkan masuk. (Ini terkadang disebut sebagai autentikasi dua faktor.)

Untuk informasi umum tentang jenis kredenSIAL yang Anda gunakan untuk mengakses Amazon Web Services, lihat KredenSIAL AWS Keamanan dalam Referensi Umum AWS panduan.

Izin dan kebijakan

Pelajari cara kerja kebijakan IAM dan temukan tip tentang cara terbaik untuk memberikan izin:

- Kebijakan dan izin di AWS Identity and Access Management – Memperkenalkan bahasa kebijakan yang digunakan untuk mendefinisikan izin. Menjelaskan bagaimana izin dapat dilampirkan ke pengguna atau grup atau, untuk beberapa produk AWS , ke sumber daya itu sendiri.
- Referensi elemen kebijakan IAM JSON – Memberikan deskripsi dan contoh setiap elemen bahasa kebijakan.
- Validasi kebijakan IAM— Temukan sumber daya untuk validasi kebijakan JSON.
- Contoh kebijakan berbasis identitas IAM— Menunjukkan contoh kebijakan untuk tugas umum di berbagai AWS produk.
- AWS Pembuat kebijakan - Buat kebijakan khusus dengan memilih produk dan tindakan dari daftar.
- Simulator Kebijakan IAM — Menguji apakah kebijakan akan mengizinkan atau menolak AWS permintaan tertentu.

Federasi dan delegasi

Anda dapat memberikan akses ke sumber daya di Anda Akun AWS untuk pengguna yang diautentikasi (masuk) di tempat lain. Ini bisa menjadi pengguna IAM di pihak lain Akun AWS (dikenal sebagai delegasi), pengguna yang diautentikasi dengan proses masuk organisasi Anda, atau pengguna dari penyedia identitas Internet seperti Login dengan Amazon, Facebook, Google, atau penyedia identitas lain yang kompatibel dengan OpenID Connect (OIDC). Dalam kasus ini, pengguna mendapatkan kredenSIAL keamanan sementara untuk mengakses AWS sumber daya.

- Tutorial IAM: Delegasikan akses di seluruh AWS akun yang menggunakan peran IAM— Memandu Anda melalui pemberian akses lintas akun ke pengguna IAM di pengguna lain. Akun AWS
- Skenario umum untuk kredensial sementara— Menjelaskan cara-cara di mana pengguna dapat digabungkan AWS setelah diautentikasi di luar. AWS

IAM dan lainnya AWS produk

Sebagian besar AWS produk terintegrasi dengan IAM sehingga Anda dapat menggunakan fitur IAM untuk membantu melindungi akses ke sumber daya dalam produk tersebut. Sumber daya berikut membahas IAM dan keamanan untuk beberapa AWS produk paling populer. Untuk daftar lengkap produk yang bekerja dengan IAM, termasuk tautan ke informasi lebih lanjut tentang masing-masing, lihat AWS layanan yang bekerja dengan IAM.

Menggunakan IAM dengan Amazon EC2

- Mengendalikan Akses ke Sumber Daya Amazon EC2 – Menjelaskan cara menggunakan fitur IAM untuk mengizinkan pengguna mengelola instans Amazon EC2, volume, dan lainnya.
- Gunakan profil instance— Menjelaskan cara menggunakan peran IAM untuk memberikan kredenSIAL dengan aman untuk aplikasi yang berjalan di instans Amazon EC2 dan yang memerlukan akses ke produk lain AWS .

Menggunakan IAM dengan Amazon S3

- Mengelola Izin Akses ke Sumber Daya Amazon S3 Anda – Membahas model kerahasiaan Amazon S3 untuk bucket dan objek yang mencakup kebijakan IAM.
- Menulis Kebijakan IAM: Berikan Akses ke User-Specific Folder di Bucket Amazon S3 — Membahas cara mengizinkan pengguna melindungi folder mereka sendiri di Amazon S3. (Untuk posting lebih lanjut tentang Amazon S3 dan IAM, pilih tag S3 di bawah judul postingan blog.)

Menggunakan IAM dengan Amazon RDS

- Menggunakan AWS Identity and Access Management (IAM) untuk Mengelola Akses ke Sumber Daya Amazon RDS — Menjelaskan cara menggunakan IAM untuk mengontrol akses ke instans database, snapshot database, dan lainnya.

- Primer tentang Izin RDS Resource-Level — Menjelaskan cara menggunakan IAM untuk mengontrol akses ke instans Amazon RDS tertentu.

Menggunakan IAM dengan Amazon DynamoDB

- Menggunakan IAM untuk Mengontrol Akses ke Sumber Daya DynamoDB — Menjelaskan cara menggunakan IAM untuk mengizinkan pengguna mengelola tabel dan indeks DynamoDB.
- Video berikut (8:55) menjelaskan cara menyediakan kontrol akses untuk item atau atribut database DynamoDB individual (atau keduanya).

  Memulai dengan Kontrol Fine-Grained Akses untuk DynamoDB

Praktik kerahasiaan umum

Temukan kiat dan panduan ahli tentang cara terbaik untuk mengamankan sumber Akun AWS daya Anda:

- Praktik Terbaik untuk Keamanan, Identitas, &, Kepatuhan — Temukan sumber daya tentang cara mengelola keamanan di seluruh Akun AWS dan produk, termasuk saran untuk arsitektur keamanan, penggunaan IAM, enkripsi dan keamanan data, dan banyak lagi.
- Manajemen Identitas dan Ak ses — AWS Well-Architected Kerangka kerja membantu Anda memahami konsep utama, prinsip desain, dan praktik terbaik arsitektur untuk merancang dan menjalankan beban kerja di cloud.
- Praktik terbaik keamanan di IAM— Menawarkan rekomendasi cara menggunakan IAM untuk membantu mengamankan sumber daya Anda Akun AWS .
- AWS CloudTrail Panduan Pengguna — Gunakan AWS CloudTrail untuk melacak riwayat panggilan API yang dilakukan AWS dan menyimpan informasi tersebut dalam file log. Ini membantu Anda menentukan pengguna dan akun mana yang mengakses sumber daya di akun Anda, kapan panggilan dilakukan, tindakan apa yang diminta, dan banyak lagi.

Sumber daya umum

Jelajahi sumber daya berikut untuk mempelajari lebih lanjut tentang IAM dan AWS.

- Informasi Produk untuk IAM - Informasi umum tentang AWS Identity and Access Management produk.

- AWS re:Post untuk AWS Identity and Access Management — Kunjungi AWS re:Post untuk membahas pertanyaan teknis terkait IAM dengan AWS komunitas.

- Kelas & Lok akarya — Tautan ke kursus berbasis peran dan khusus, selain laboratorium mandiri untuk membantu mempertajam AWS keterampilan Anda dan mendapatkan pengalaman praktis.
- AWS Pusat Pengembang — Jelajahi tutorial, unduh alat, dan pelajari tentang acara AWS pengembang.
- AWS Alat Pengembang — Tautan ke alat pengembang, SDK, toolkit IDE, dan alat baris perintah untuk mengembangkan dan mengelola AWS aplikasi.
- Pusat Sumber Daya Mem ulai — Pelajari cara mengatur Akun AWS, bergabung dengan AWS komunitas, dan meluncurkan aplikasi pertama Anda.
- Hands-On Tutorial - Ikuti tutorial langkah demi langkah untuk meluncurkan aplikasi pertama Anda AWS.
- AWS Whitepapers — Tautan ke daftar lengkap AWS whitepaper teknis, yang mencakup topik-topik seperti arsitektur, keamanan, dan ekonomi dan ditulis oleh Arsitek Solusi atau pakar teknis lainnya. AWS
- AWS Dukungan Center — Hub untuk membuat dan mengelola AWS Dukungan kasus Anda. Juga termasuk tautan ke sumber daya bermanfaat lainnya, seperti forum, FAQ teknis, status kesehatan layanan, dan AWS Trusted Advisor.
- Dukungan— Halaman web utama untuk informasi tentang Dukungan, saluran dukungan respons cepat satu-satu untuk membantu Anda membangun dan menjalankan aplikasi di cloud.
- Hubungi Kami – Titik kontak pusat untuk pertanyaan tentang tandaihan AWS , akun, peristiwa, penyalahgunaan, dan masalah lainnya.
- AWS Ketentuan Situs — Informasi terperinci tentang hak cipta dan merek dagang kami; akun Anda, lisensi, dan akses situs; dan topik lainnya.

Memanggil API IAM menggunakan permintaan kueri HTTP

Daftar Isi

- Titik akhir
- HTTPS diperlukan
- Menandatangani permintaan API IAM

Anda dapat mengakses IAM dan AWS STS layanan secara terprogram menggunakan Query API. Permintaan API Kueri adalah permintaan HTTPS yang harus berisi parameter `Action` untuk menunjukkan tindakan yang akan dilakukan. IAM dan AWS STS mendukung permintaan GET dan POST untuk semua tindakan. Artinya, API tidak mewajibkan Anda menggunakan GET untuk beberapa tindakan dan POST untuk yang lainnya. Namun, permintaan GET harus memenuhi ukuran batas dari URL; meskipun batas ini bergantung pada browser, batas umumnya adalah 2048 byte. Oleh karena itu, untuk permintaan API Kueri yang memerlukan ukuran lebih besar, Anda harus menggunakan permintaan POST.

Responsnya adalah dokumen XML. Untuk detail tentang respons, lihat halaman tindakan individual di Referensi API IAM atau Referensi AWS Security Token Service API.

> **Tip**
>
> Alih-alih melakukan panggilan langsung ke operasi IAM atau AWS STS API, Anda dapat menggunakan salah satu SDK AWS . AWS SDK terdiri dari pustaka dan kode sampel untuk berbagai bahasa pemrograman dan platform (Java, Ruby, .NET, iOS, Android, dll.). SDK menyediakan cara mudah untuk membuat akses terprogram ke IAM dan. AWS Misalnya, SDK menangani tugas seperti menandatangani permintaan secara kriptografis (lihat di bawah ini), mengelola kesalahan, dan mencoba kembali permintaan secara otomatis. Untuk informasi tentang AWS SDK, termasuk cara mengunduh dan menginstalnya, lihat halaman Alat untuk Amazon Web Services.

Untuk detail tentang tindakan dan kesalahan API, lihat Referensi API IAM atau Referensi AWS Security Token Service API.

Titik akhir

IAM dan AWS STS masing-masing memiliki titik akhir global tunggal:

- (IAM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Important**
>
> AWS STS juga mendukung pengiriman permintaan ke titik akhir regional selain titik akhir global. AWS merekomendasikan penggunaan titik akhir regional alih-alih titik akhir global untuk mengurangi latensi, membangun redundansi, dan meningkatkan validitas token sesi. Sebelum Anda dapat menggunakan AWS STS di Wilayah, Anda harus terlebih dahulu mengaktifkan STS di Wilayah itu untuk Anda Akun AWS. Untuk informasi selengkapnya tentang mengaktifkan Wilayah tambahan untuk AWS STS, lihatKelola AWS STS dalam sebuah Wilayah AWS.

Untuk informasi selengkapnya tentang AWS titik akhir dan Wilayah untuk semua layanan, lihat T itik akhir layanan dan kuota di. Referensi Umum AWS

HTTPS diperlukan

Karena API Kueri mengembalikan informasi sensitif seperti kredensial keamanan, Anda harus menggunakan HTTPS dengan semua permintaan API.

Menandatangani permintaan API IAM

Permintaan harus ditandatangani menggunakan access key ID dan secret access key. Kami sangat menyarankan agar Anda tidak menggunakan Pengguna root akun AWS kredenSIAL Anda untuk pekerjaan sehari-hari dengan IAM. Anda dapat menggunakan kredenSIAL untuk pengguna IAM atau Anda dapat menggunakan AWS STS untuk menghasilkan kredenSIAL keamanan sementara.

Untuk menandatangani permintaan API Anda, sebaiknya gunakan AWS Signature Version 4. Untuk informasi tentang menggunakan Tanda Tangan Versi 4, kunjungi Proses Penandatanganan Tanda Tangan Versi 4 di Referensi Umum AWS .

Jika Anda perlu menggunakan Signature Versi 2, informasi tentang menggunakan Signature Versi 2 tersedia di Referensi Umum AWS.

Untuk informasi selengkapnya, lihat berikut ini:

- AWS KredenSIAL Keamanan. Memberikan informasi umum tentang jenis kredenSIAL yang digunakan untuk mengakses AWS.
- Praktik terbaik keamanan di IAM. Menyajikan daftar saran untuk menggunakan layanan IAM untuk membantu mengamankan AWS sumber daya Anda.
- Kredensial keamanan sementara di IAM. Menjelaskan cara membuat dan menggunakan kredensial keamanan sementara.

Riwayat dokumen untuk IAM

Tabel berikut menjelaskan pembaruan dokumentasi utama untuk IAM.

Perubahan	Deskripsi	Tanggal
job_workflow_ref diterima sebagai klaim pelingkupan alternatif untuk penyedia Actions OIDC GitHub	Untuk penyedia GitHub Actions OIDC, IAM sekarang menerima `sub` klaim atau `job_workflow_ref` klaim sebagai kontrol penyedia identitas dalam kebijakan kepercayaan peran.	12 Mei 2026
Kunci konteks kondisi IAM untuk API kredensia khusus layanan	IAM sekarang mendukung dua kunci konteks kondisi baru untuk mengontrol akses ke API kredensia khusus layanan: dan. `iam:ServiceSpecificCredentialAgeDays` `iam:ServiceSpecificCredentialServiceName` Kunci ini memungkinkan Anda membatasi pembuatan dan pengelolaan kredenSIAL khusus layanan berdasarkan pengaturan kedaluwarsa dan layanan yang diizinkan AWS . • `iam:ServiceSpecificCredentialAgeDays` • `iam:ServiceSpecificCredentialServiceName`	4 September 2025

Kunci kondisi titik akhir VPC untuk kontrol perimeter jaringan	IAM sekarang mendukung tiga kunci konteks kondisi AWS global baru untuk menerapkan kontrol perimeter jaringan yang dapat diskalakan:`aws:VpceAccount` ,`aws:Vpce0rgID` , dan`aws:Vpce0rgPaths` . Kunci ini membantu memastikan permintaan datang melalui titik akhir VPC yang dimiliki oleh akun, organisasi, atau unit organisasi tertentu, secara otomatis menskalakan dengan penggunaan titik akhir VPC Anda tanpa memerlukan pembaruan kebijakan saat Anda membuat titik akhir baru. Untuk informasi selengkap nya, lihat Men etapkan pagar pembatas izin menggunakan perimeter data.	Agustus 28, 2025

Tutorial federasi IAM SAML CloudFormation	IAM menambahkan tutorial baru untuk membuat penyedia identitas SAML (IdPs) dan peran federasi menggunakan. CloudFormation • Buat SAML IdP dengan CloudFormation • Buat peran gabungan SAML dengan CloudFormation • Buat SAML IdP dan peran federasi dengan CloudFormation	Agustus 25, 2025
Kontrol penyedia OIDC bersama tambahan untuk penyedia OIDC bersama	IAM sekarang menyertakan titik akhir global Amazon Cognito, Azure Sentinel, Pulumi Cloud, dan Vercel dalam daftar penyedia identitas OIDC bersama yang memerlukan evaluasi eksplisit klaim spesifik dalam Token Web JSON (JWT).	Agustus 1, 2025

IAM Access Analyzer menambahkan penganalisis akses internal	IAM Access Analyzer membantu Anda mengidentifikasi prinsipal mana dalam organisasi atau akun Anda yang memiliki akses ke sumber daya penting bisnis yang dipilih. Penganalisis akses internal mendukung penerapan prinsip hak istimewa terendah dengan memastikan bahwa sumber daya yang Anda tentukan hanya dapat diakses oleh prinsipal yang dituju dalam AWS organisasi atau akun Anda.	Juni 16, 2025
Kontrol penyedia identitas untuk penyedia OIDC bersama	IAM sekarang memerlukan evaluasi eksplisit klaim spesifik dalam Token Web JSON (JWT) untuk penyedia identitas OIDC bersama yang diakui. Kontrol keamanan ini memastikan bahwa hanya identitas resmi dari organisasi yang dituju yang dapat mengambil peran dan mengakses AWS sumber daya.	Juni 6, 2025
`IAMUserChangePassword` — Menambahkan izin	IAM menambahkan izin `IAMUserChangePassword` untuk mengizinkan pengguna yang ditentukan dalam jalur.	28 Mei 2025

AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer ditambahkan iam:GetAccountAuthorizationDetails ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	12 Mei 2025
AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan titik akses bucket direktori Amazon S3 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Maret 31, 2025
Dukungan titik akhir dual-stack IAM	IAM sekarang menyediakan dukungan titik akhir dual-stack yang ditingkatkan yang memungkinkan klien berkomunikasi dengan IAM menggunakan alamat IPv4 atau IPv6.	Maret 20, 2025
Pembaruan logika evaluasi kebijakan	Dokumentasi yang disempurnakan untuk logika evaluasi kebijakan, termasuk diagram alur yang ditingkatkan dan penjelasan yang lebih jelas tentang cara AWS mengevaluasi kebijakan untuk menentukan apakah akan mengizinkan atau menolak permintaan.	Januari 30, 2025
IAMDeleteRootUserCredentials — Izin yang dihapus	IAM menghapus iam:DeleteVirtualMFADevice izin dari kebijakan terkelola.	Januari 7, 2025

`AccessAnalyzerServiceRolePolicy` — Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengaturan akun Amazon ECR dan kebijakan registri ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Desember 10, 2024
IAM Access Analyzer menambahkan konfigurasi akses	IAM Access Analyzer menambahkan dukungan untuk mengonfigurasi penganalisis untuk mengubah ruang lingkup, pengguna IAM Akun AWS, dan peran yang menghasilkan temuan.	14 November 2024
Mengelola akses root secara terpusat untuk akun anggota	Anda sekarang dapat mengelola kredenSIAL pengguna root yang memiliki hak istimewa di seluruh akun anggota AWS Organizations dengan akses root terpusat. Amankan secara terpusat kredensi pengguna root dari Akun AWS penggunaan terkelola Anda AWS Organizations untuk menghapus dan mencegah pemulihan kredensi pengguna root dan akses dalam skala besar. • Praktik terbaik pengguna root	14 November 2024

AWS pembaruan kebijakan terkelola - Kebijakan baru	IAM menambahkan dua kebijakan baru untuk mencakup izin untuk sesi pengguna root istimewa yang dapat Anda mulai setelah Anda memusatkan akses pengguna root untuk akun anggota di organisasi Anda. • IAMAuditRootUserCredentials • IAMCreateRootUserPassword • IAMDeleteRootUserCredentials • S3UnlockBucketPolicy • SQSUnlockQueuePolicy	14 November 2024
Dukungan untuk kebijakan kontrol AWS Organizations sumber daya (RCP)	Gunakan kebijakan kontrol AWS Organizations sumber daya (RCP) untuk menentukan izin maksimum untuk sumber daya dalam akun di organisasi atau unit organisasi (OU) Anda. RCP membatasi izin yang dapat diberikan kebijakan berbasis identitas dan berbasis sumber daya ke sumber daya di akun dalam organisasi Anda.	November 13, 2024

AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengguna IAM dan tag peran ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Oktober 29, 2024
Peningkatan dukungan enkripsi SAML	Dokumentasi yang disempurnakan untuk dukungan enkripsi SAML di penyedia IAM SAML, termasuk panduan pemecahan masalah yang ditingkatkan dan klarifikasi tentang kompatibilitas layanan.	5 Juni 2024
AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil informasi tentang pengguna IAM dan kebijakan peran ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	30 Mei 2024
Dukungan enkripsi untuk penyedia identitas SAML	Penyedia IAM SAML sekarang mendukung pernyataan terenkripsi dalam respons SAML dari IdP eksternal Anda. Untuk memahami cara kerja enkripsi dengan federasi IAM SAML, lihat Menggunakan SAML-based federasi untuk akses API.	Februari 4, 2024

AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan dukungan untuk izin untuk mengambil status saat ini dari akses publik blok untuk snapshot Amazon EC2 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	23 Januari 2024
AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan aliran dan tabel DynamoDB ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	Januari 11, 2024
AccessAnalyzerServiceRolePolicy — Menambahkan izin	IAM Access Analyzer menambahkan bucket direktori Amazon S3 ke izin tingkat layanan. AccessAnalyzerServiceRolePolicy	1 Desember 2023
IAMAccessAnalyzerReadOnlyAccess — Menambahkan izin	IAM Access Analyzer menambahkan izin IAMAccessAnalyzerReadOnlyAccess untuk memungkinkan Anda memeriksa apakah pembaruan kebijakan Anda memberikan akses tambahan. Izin ini diperlukan oleh IAM Access Analyzer untuk melakukan pemeriksaan kebijakan pada kebijakan Anda.	26 November 2023

IAM Access Analyzer menambahkan penganalisis akses yang tidak digunakan	IAM Access Analyzer menyederhanakan pemeriksaan akses yang tidak digunakan untuk memandu Anda menuju hak istimewa yang paling sedikit. IAM Access Analyzer terus menganalisis akun Anda untuk mengidentifikasi akses yang tidak digunakan dan membuat dasbor terpusat dengan temuan.	26 November 2023
IAM Access Analyzer menambahkan pemeriksaan kebijakan kustom	IAM Access Analyzer sekarang menyediakan pemeriksaan kebijakan khusus untuk memvalidasi bahwa kebijakan IAM mematuhi standar keamanan Anda sebelum penerapan.	26 November 2023
`AccessAnalyzerServiceRolePolicy` — Menambahkan izin	IAM Access Analyzer menambahkan tindakan IAM ke izin tingkat layanan AccessAnalyzerServiceRolePolicy untuk mendukung tindakan berikut: • Mendaftarkan entitas untuk kebijakan • Menghasilkan detail layanan yang terakhir diakses • Informasi kunci akses daftar	26 November 2023

Tindakan terakhir diakses informasi dan dukungan pembuatan kebijakan untuk lebih dari 60 layanan dan tindakan tambahan	IAM sekarang mendukung informasi tindakan yang terakhir diakses dan menghasilkan kebijakan dengan informasi tingkat tindakan untuk lebih dari 60 layanan tambahan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan tersedia.	1 November 2023
Tindakan terakhir diakses dukungan informasi untuk lebih dari 140 layanan	IAM sekarang menyediakan informasi tindakan terakhir diakses untuk lebih dari 140 layanan, bersama dengan daftar tindakan yang informasi terakhir diakses tindakan tersedia.	14 September 2023
Dukungan untuk beberapa perangkat otentikasi multi-faktor (MFA) untuk pengguna root dan pengguna IAM	Sekarang Anda dapat menambahkan hingga delapan perangkat MFA per pengguna, termasuk kunci keamanan FIDO, kata sandi satu kali berbasis waktu perangkat lunak (TOTP) dengan aplikasi otentikator virtual, atau token TOTP perangkat keras.	16 November 2022

Dukungan IAM Access Analyzer untuk jenis sumber daya baru	IAM Access Analyzer menambahkan dukungan untuk jenis sumber daya berikut: • Cuplikan volume Amazon EBS • Repositori Amazon ECR • Sistem file Amazon EFS • Cuplikan Amazon RDS DB • Snapshot cluster Amazon RDS DB • Topik Amazon SNS	25 Oktober 2022
Penghentian dan pembaruan U2F WebAuthn/FIDO	Menghapus penyebutan U2F sebagai opsi MFA dan menambahkan informasi tentang WebAuthn, FIDO2, dan kunci keamanan FIDO.	31 Mei 2022
Pembaruan ketahanan di IAM	Menambahkan informasi tentang mempertahankan akses ke kredenSIAL IAM ketika suatu peristiwa mengganggu komunikasi di antara. Wilayah AWS	Mei 16, 2022

Kunci kondisi global baru untuk sumber daya	Anda sekarang dapat mengontrol akses ke sumber daya berdasarkan akun, Unit Organisasi (OU), atau organisasi AWS Organizat ions yang berisi sumber daya Anda. Anda dapat menggunak an kunci kondisi `aws:Resou rceAccount` `aws:Resou rceOrgID` ,, dan `aws:Resou rceOrgPaths` global dalam kebijakan IAM.	27 April 2022
Contoh kode untuk IAM menggunakan SDK AWS	Menambahkan contoh kode yang menunjukkan cara menggunakan IAM dengan AWS perangkat pengemban gan perangkat lunak (SDK). Contoh-contoh dibagi menjadi kutipan kode yang menunjukk an cara memanggil fungsi layanan individual dan contoh yang menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.	7 April 2022
Pembaruan bagan alir logika evaluasi kebijakan	Pembaruan bagan alir logika evaluasi kebijakan dan teks terkait di bagian Men entukan apakah permintaan diizinkan atau ditolak dalam akun.	17 November 2021

Pembaruan praktik terbaik keamanan	Menambahkan informasi tentang membuat pengguna administratif alih-alih menggunakan kredentif pengguna root, menghapus praktik terbaik menggunakan grup IAM untuk menetapkan izin kepada pengguna IAM, dan mengklarifikasi kapan harus menggunakan kebijakan terkelola alih-alih kebijakan sebaris.	5 Oktober 2021
Pembaruan topik logika evaluasi kebijakan untuk kebijakan berbasis sumber daya	Menambahkan informasi tentang dampak kebijakan berbasis sumber daya dan jenis utama yang berbeda dalam akun yang sama.	5 Oktober 2021
Pembaruan untuk kunci kondisi bernilai tunggal dan multinilai	Perbedaan antara kunci kondisi bernilai tunggal dan multinilai sekarang dijelaskan secara lebih rinci. Jenis nilai ditambahkan ke setiap kunci konteks kondisi AWS global.	30 September 2021
Penganalisis Akses IAM mendukung Titik Multi-Region Akses Amazon S3	IAM Access Analyzer mengidentifikasi bucket Amazon S3 yang memungkinkan akses publik dan lintas akun, termasuk yang menggunakan Amazon S3 Multi-Region Access Points.	2 September 2021
AWS pembaruan kebijakan terkelola - Memperbarui ke kebijakan yang ada	IAM Access Analyzer memperbarui kebijakan terkelola yang AWS ada.	2 September 2021

Lebih banyak layanan yang didukung untuk pembuatan kebijakan tingkat tindakan	IAM Access Analyzer dapat menghasilkan kebijakan IAM dengan informasi aktivitas akses tingkat tindakan untuk layanan tambahan. AWS	Agustus 24, 2021
Buat kebijakan IAM untuk jalur lintas akun	Anda sekarang dapat menggunakan Penganalisis Akses IAM untuk menghasil kan kebijakan berbutir halus berdasarkan aktivitas akses Anda menggunakan AWS CloudTrail jejak di akun yang berbeda, misalnya, jejak terpusat. AWS Organizations	18 Agustus 2021

Pemeriksaan kebijakan IAM Access Analyzer tambahan	IAM Access Analyzer memperpanjang validasi kebijakan dengan menambahkan pemeriksaan kebijakan baru yang memvalidasi kondisi yang disertakan dalam kebijakan IAM. Pemeriksaan ini menganalisis blok syarat dalam pernyataan kebijakan Anda dan melaporkan peringatan, kesalahan, dan saran keamanan beserta rekomendasi yang dapat ditindaklanjuti. IAM Access Analyzer menambahkan pemeriksaan kebijakan berikut: • Kesalahan — Format utama layanan tidak valid • Kesalahan - Kunci tag hilang dalam kondisi • Peringatan Keamanan — Tolak NotAction dengan kunci kondisi tag yang tidak didukung untuk layanan • Peringatan Keamanan — Tolak dengan kunci kondisi tag yang tidak didukung untuk layanan • Peringatan Keamanan - Kunci kondisi yang dipasangkan hilang • Saran - Izinkan NotAction dengan kunci kondisi tag	29 Juni 2021

	yang tidak didukung untuk layanan • Saran - Izinkan dengan kunci kondisi tag yang tidak didukung untuk layanan	
Tindakan terakhir diakses dukungan untuk lebih banyak layanan	Anda sekarang dapat melihat informasi tindakan yang terakhir diakses di konsol IAM tentang kapan terakhir kali prinsipal IAM menggunakan tindakan untuk layanan berikut: Amazon EC2, IAM, Lambda, dan tindakan manajemen Amazon S3. Anda juga dapat menggunakan AWS API AWS CLI atau untuk mengambil laporan data. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan kebijakan IAM Anda agar lebih mematuhi prinsip hak istimewa terkecil.	19 April 2021
Memantau dan mengontrol tindakan yang diambil dengan peran yang diasumsikan	Administrator dapat mengonfigurasi peran IAM untuk mewajibkan identitas meneruskan identitas sumber, yang masuk AWS CloudTrail. Meninjau informasi identitas sumber membantu administrator menentukan siapa atau apa yang dilakukan tindakan dengan sesi peran diasumsikan.	13 April, 2021

Buat kebijakan IAM berdasarkan aktivitas akses	Anda sekarang dapat menggunakan Penganalisis Akses IAM untuk menghasilkan kebijakan berbutir halus berdasarkan aktivitas akses yang ditemukan di situs Anda. AWS CloudTrail	7 April 2021
Pemeriksaan kebijakan IAM Access Analyzer	IAM Access Analyzer kini menyediakan lebih dari 100 pemeriksaan kebijakan dengan rekomendasi yang dapat ditindaklanjuti selama pembuatan kebijakan.	16 Maret 2021
Opsi validasi kebijakan yang diperluas	Validasi kebijakan yang diperluas tersedia di konsol IAM, AWS API, dan AWS CLI menggunakan pemeriksaan kebijakan di IAM Access Analyzer untuk membantu Anda membuat kebijakan JSON yang aman dan fungsional.	15 Maret 2021
Menandai sumber daya IAM	Anda sekarang dapat menandai sumber daya IAM tambahan menggunakan pasangan nilai kunci tag.	11 Februari 2021
Kebijakan kata sandi default untuk pengguna IAM	Jika Anda tidak menetapkan kebijakan kata sandi khusus untuk Anda Akun AWS, kata sandi pengguna IAM sekarang harus memenuhi kebijakan AWS kata sandi default.	18 November 2020

Halaman tindakan, sumber daya, dan kunci kondisi untuk AWS layanan telah dipindahkan	Setiap AWS layanan dapat menentukan tindakan, sumber daya, dan kunci konteks kondisi untuk digunakan dalam kebijakan IAM. Anda sekarang dapat menemukan daftar AWS layanan dan tindakan, sumber daya, dan kunci konteks kondisi mereka di Referensi Otorisasi Layanan.	16 November 2020
Durasi sesi peran pengguna IAM lebih lama	Pengguna IAM sekarang dapat memiliki durasi sesi peran yang lebih lama saat beralih peran di Konsol Manajemen AWS, mengurangi interupsi karena kedaluwarsa sesi. Pengguna diberi durasi sesi maksimum yang ditetapkan untuk peran tersebut, atau waktu yang tersisa di sesi pengguna IAM, mana pun yang lebih sedikit.	24 Juli 2020
Gunakan Kuota Layanan untuk meminta peningkatan cepat untuk entitas IAM	Anda dapat meminta peningkatan kuota untuk kuota IAM yang dapat disesuaikan menggunakan konsol Kuota Layanan. Sekarang, beberapa peningkatan disetujui secara otomatis di Kuota Layanan dan tersedia di akun Anda dalam beberapa menit. Permintaan yang lebih besar diajukan ke AWS Dukungan.	25 Juni 2020

Informasi terakhir yang diakses di IAM sekarang termasuk tindakan manajemen Amazon S3	Selain informasi layanan yang terakhir diakses, Anda sekarang dapat melihat informasi di konsol IAM tentang saat terakhir kali penanggung jawab IAM menggunakan tindakan Amazon S3. Anda juga dapat menggunakan AWS API AWS CLI atau untuk mengambil laporan data. Laporan ini mencakup informasi tentang layanan dan tindakan yang diizinkan yang terakhir kali dilakukan untuk mengakses dan kapan dilakukan oleh penanggung jawab tersebut. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempurnakan kebijakan IAM Anda agar lebih mematuhi prinsip hak istimewa terkecil.	3 Juni 2020

Penambahan Bab Keamanan	Bab keamanan membantu Anda memahami cara mengonfigurasi IAM dan AWS STS memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelaja ri cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya IAM Anda.	29 April 2020
sts: RoleSessionName	Sekarang Anda dapat menyusun kebijakan yang memberikan izin berdasark an nama sesi yang ditentukan oleh penanggung jawab saat mengambil peran.	21 April 2020
AWS pembaruan halaman masuk	Saat Anda masuk di halaman AWS masuk utama, Anda sekarang dapat memilih untuk masuk sebagai Pengguna root akun AWS atau pengguna IAM. Ketika Anda melakukan nya, label pada halaman menunjukkan apakah Anda harus memberikan alamat email pengguna root atau informasi pengguna IAM Anda. Dokumentasi ini mencakup tangkapan layar yang diperbarui untuk membantu Anda memahami halaman masuk AWS .	4 Maret 2020

AWS: viaawsService dan aws: kunci kondisi CalledVia	Sekarang Anda dapat menyusun kebijakan untuk membatasi apakah layanan dapat membuat permintaan atas nama penanggung jawab IAM (pengguna atau peran). Saat penanggung jawab mengajukan permintaan ke layanan AWS , layanan tersebut mungkin menggunakan kredensial penanggung jawab untuk melakukan permintaan selanjutnya ke layanan lain. Gunakan `aws:ViaAWSService` kunci kondisi yang harus dicocokkan jika layanan apa pun membuat permintaan menggunakan kredensial penanggung jawab. Gunakan `aws:CalledVia` kunci kondisi yang harus dicocokkan jika layanan tertentu membuat permintaan menggunakan kredensial penanggung jawab.	20 Februari 2020
Simulator kebijakan menambahkan dukungan untuk batas izin	Sekarang Anda dapat menguji efek batas izin pada entitas IAM dengan simulator kebijakan IAM.	23 Januari 2020

Cross-account evaluasi kebijakan	Sekarang Anda dapat mempelajari cara AWS mengevaluasi kebijakan untuk akses lintas akun. Hal ini terjadi ketika sumber daya di akun tepercaya termasuk kebijakan berbasis sumber daya yang memungkinkan penanggung jawab di akun lain mengakses sumber daya tersebut. Permintaan harus diizinkan di kedua akun.	2 Januari 2020
Tag sesi	Sekarang Anda dapat menyertakan tanda saat Anda mengambil peran atau menggabungkan pengguna di AWS STS. Saat Anda melakukan operasi `AssumeRole` atau `GetFederationToken`, Anda dapat lulus tanda sesi sebagai atribut. Saat Anda melakukan `AssumeRoleWithWebIdentity` operasi `AssumeRoleWithSAML` atau, Anda dapat meneruskan atribut dari identitas perusahaan Anda ke AWS.	22 November 2019

Kontrol akses untuk kelompok Akun AWS masuk AWS Organizations	Anda sekarang dapat mereferensikan unit organisasi (OU) dari AWS Organizations kebijakan IAM. Jika Anda menggunakannya AWS Organizations untuk mengatur akun Anda menjadi OU, Anda dapat meminta prinsipal milik OU tertentu sebelum memberikan akses ke sumber daya Anda. Prinsip termasuk Pengguna root akun AWS, pengguna IAM dan peran IAM. Untuk melakukannya, tentukan jalur OU di kunci kondisi `aws:PrincipalOrgPaths` di kebijakan Anda.	20 November 2019
Peran terakhir digunakan	Sekarang Anda dapat melihat tanggal, waktu, dan Wilayah tempat peran terakhir digunakan. Informasi ini juga membantu Anda mengidentifikasi peran yang tidak digunakan di akun Anda. Anda dapat menggunakan Konsol Manajemen AWS, AWS CLI dan AWS API untuk melihat informasi tentang kapan peran terakhir digunakan.	19 November 2019

Perbarui ke halaman kunci konteks kondisi global	Sekarang Anda dapat mempelajari kapan setiap kunci kondisi global disertakan dalam konteks permintaan. Anda juga dapat menuju ke setiap kunci dengan lebih mudah menggunakan daftar isi (TOC). Informasi di laman tersebut membantu Anda menulis kebijakan yang lebih akurat. Misalnya, jika karyawan Anda menggunakan federasi dengan peran IAM, Anda harus menggunakan kunci `aws:userId` dan bukan kunci `aws:userName`. Kunci `aws:userName` hanya berlaku untuk pengguna IAM, bukan peran.	6 Oktober 2019

ABAC ke AWS	Pelajari cara kerja kontrol akses berbasis atribut (ABAC) dalam AWS menggunakan tag, dan bagaimana perbandingannya dengan model AWS otorisasi tradisional. Gunakan tutorial ABAC untuk mempelajari cara membuat dan menguji kebijakan yang memungkinkan peran IAM dengan tag penanggung jawab mengakses sumber daya dengan tag yang sesuai. Strategi ini memungkinkan individu untuk melihat atau mengedit hanya AWS sumber daya yang diperlukan untuk pekerjaan mereka.	3 Oktober 2019
AWS STS GetAccessKeyInfo operasi	Anda dapat meninjau kunci AWS akses dalam kode Anda untuk menentukan apakah kunci tersebut berasal dari akun yang Anda miliki. Anda dapat meneruskan ID kunci akses menggunakan `aws sts get-access-key-info` AWS CLI perintah atau operasi `GetAccessKeyInfo` AWS API.	24 Juli 2019

Melihat informasi AWS Organizations layanan yang terakhir diakses di IAM	Anda sekarang dapat melihat informasi layanan yang terakhir diakses untuk AWS Organizations entitas atau kebijakan di AWS Organizat ions bagian konsol IAM. Anda juga dapat menggunakan AWS API AWS CLI atau untuk mengambil laporan data. Data ini mencakup informasi tentang layanan yang diizinkan yang terakhir kali dicoba diakses AWS Organizations oleh kepala sekolah dalam akun dan kapan. Anda dapat menggunakan informasi ini untuk mengidentifikasi izin yang tidak perlu sehingga Anda dapat menyempur AWS Organizations nakan kebijakan agar lebih mematuhi prinsip hak istimewa paling sedikit.	20 Juni 2019
Menggunakan kebijakan terkelola sebagai kebijakan sesi	Sekarang Anda dapat memberikan hingga 10 ARN kebijakan terkelola saat Anda mengambil peran. Hal ini memungkinkan Anda untuk membatasi izin kredensial sementara peran.	7 Mei 2019

AWS STS Kompatibilitas wilayah token sesi untuk titik akhir global	Sekarang, Anda dapat memilih apakah menggunakan token titik akhir global versi 1 atau versi 2. Token versi 1 hanya berlaku di AWS Wilayah yang tersedia secara default. Token ini tidak akan bekerja di Wilayah yang diaktifkan secara manual, seperti Asia Pacific (Hong Kong). Token Versi 2 valid di semua Wilayah. Namun, token versi 2 lebih panjang dan mungkin memengaruhi sistem tempat Anda menyimpan token untuk sementara waktu.	26 April 2019
Izinkan mengaktifkan dan menonaktifkan AWS wilayah	Sekarang Anda dapat membuat kebijakan yang memungkinkan administrator untuk mengaktifkan dan menonaktifkan Wilayah Asia Pasifik (Hong Kong) (ap-east-1).	24 April 2019

IAM pengguna halaman kredentif keamanan saya	Pengguna IAM sekarang dapat mengelola semua kredensial mereka sendiri di laman Kredensial Keamanan Saya. Konsol Manajemen AWS Halaman ini menampilk an informasi akun seperti ID akun dan ID pengguna kanonik. Pengguna juga dapat melihat dan mengedit kata sandi mereka sendiri, kunci akses, X.509 sertifikat, kunci SSH, dan kredenSIAL Git.	24 Januari 2019
API Penganalisis Akses	Anda sekarang dapat menggunakan AWS CLI dan AWS API untuk melihat informasi layanan yang terakhir diakses.	7 Desember 2018
Menandai pengguna dan peran IAM	Sekarang, Anda dapat menggunakan tag IAM untuk menambahkan atribut khusus ke identitas (pengguna atau peran IAM) menggunakan pasangan nilai kunci tag. Anda juga dapat menggunakan tag untuk mengendalikan akses identitas ke sumber daya atau untuk mengontrol tag apa yang dapat dipasang ke sebuah identitas.	14 November 2018

Kunci keamanan U2F	Sekarang Anda dapat menggunakan kunci keamanan U2F sebagai opsi autentikasi multi-faktor (MFA) saat masuk ke Konsol Manajemen AWS.	25 September 2018
Dukungan untuk titik akhir VPC Amazon VPC	Anda sekarang dapat membuat koneksi pribadi antara VPC Anda dan AWS STS di Wilayah AS Barat (Oregon).	31 Juli 2018
Batas izin	Fitur baru memudahkan pemberian kemampuan kepada karyawan tepercaya untuk mengelola izin IAM tanpa memberikan akses administratif IAM penuh.	12 Juli 2018
aws: PrincipalOrg ID	Kunci kondisi baru menyediakan cara yang lebih mudah untuk mengontrol akses ke AWS sumber daya dengan menentukan AWS organisasi prinsipal IAM.	17 Mei 2018
aws: RequestedRegion	Kunci kondisi baru menyediakan cara yang lebih mudah untuk menggunakan kebijakan IAM untuk mengontrol akses ke Wil AWS ayah.	25 April 2018
Peningkatan durasi sesi untuk peran IAM	Peran IAM sekarang dapat memiliki durasi sesi 12 jam.	28 Maret 2018

Alur kerja pembuatan peran yang diperbarui	Alur kerja baru meningkatkan proses pembuatan hubungan kepercayaan dan melampirkan izin ke peran.	8 September 2017
Akun AWS proses masuk	Peng AWS alaman masuk yang diperbarui memungkin kan pengguna root dan pengguna IAM menggunakan tautan Mas uk ke Konsol di halaman Konsol Manajemen AWS beranda.	25 Agustus 2017
Contoh kebijakan IAM	Pembaruan dokumentasi menampilkan lebih dari 30 kebijakan contoh.	2 Agustus 2017
Praktik terbaik IAM	Informasi yang ditambahkan ke bagian Pengguna konsol IAM memudahkan untuk mengikuti praktik terbaik IAM.	5 Juli 2017
Sumber daya Penskalaan Otomatis	Resource-level izin dapat mengontrol akses ke dan izin untuk sumber daya Penskalaa n Otomatis.	16 Mei 2017
Amazon RDS untuk database MySQL dan Amazon Aurora	Administrator database dapat mengaitkan pengguna database dengan pengguna dan peran IAM dan dengan demikian mengelola akses pengguna ke semua AWS sumber daya dari satu lokasi.	24 April 2017

Service-linked peran	Service-linked peran menyediakan cara yang lebih mudah dan lebih aman untuk mendelegasikan izin ke AWS layanan.	19 April 2017
Ringkasan kebijakan	Ringkasan kebijakan baru memudahkan untuk memahami izin di kebijakan IAM.	23 Maret 2017

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.